TEA SPILLERS | Billing System

A System Development Documentation
Submitted to the Faculty
of the Department of Information Technology
Cavite State University - Imus Campus
City of Imus, Cavite

In partial fulfillment
of the requirements in the subject,
DCIT 23 Computer Programming 2

Fabian, Meg Angeline V.
Koa, Kristine L.
Murillo, Pamela T.
Palima, Ginna V.
Velasco, Hazel Mae C.

July 10, 2023

System Description

This GUI application called Tea Spillers is designed to provide a user-friendly interface for customers visiting a coffee shop. With this application, customers can easily enter their details, browse the available products, calculate the total price of their order, and generate a receipt.

System Features

Here are the features of Tea Spillers Billing System:

1. Login and Sign-Up Functionality:
   - Users can log in using predefined usernames and passwords.
   - Users can sign up to create a new account.

2. Menu Section:
   - Display names and prices of different coffee items.
   - Prices are read-only and cannot be directly edited.

3. Order Section:
   - You can click the images of the products to add quantity of different drinks.
   - It has minus buttons, trash buttons, and billing area.

4. Receipt Section:
   - Display the final bill or order details.
   - Supports scrolling for longer receipts.

5. Action Buttons:
   - Calculate Total: Calculate the total bill amount based on the entered quantities.
   - Receipt: Generate and display the bill or order details in the receipt area.
   - Print: Print the receipt.
   - Reset: Clear all entered quantities and the receipt area.
   - Exit: Close the application or go back to the login screen.

These features collectively provide functionality for logging in, choosing quantities of

desserts and drinks, calculating the total bill, generating a receipt, and performing actions such as printing and resetting the order.

System Information

The system sets up a GUI application for a coffee shop where customers can enter their details, select products, calculate the total price, and print a receipt. The receipt can be displayed on the screen for the customer to review and, if desired, printed for their convenience.

System Future Suggestion

If you have any recommendations or face any problems while using the system, consider offering feedback to the system's creators or support team. Your suggestions can help us enhance the system for all users in the future.

System Codes

```
from tkinter import *
from tkinter import messagebox
from PIL import Image, ImageTk
from fpdf import FPDF

root = Tk()
root.title('Tea Spillers | Sign in')
root.geometry("951x500+300+200")
root.configure(bg="#fff")
root.resizable(False, False)

img = PhotoImage(file='logongteaspillers.png')
Label(root, image=img, bg='white').place(x=10, y=10)

coffee_shop = 'Tea Spillers'
shop_address = 'Dream Villa Imus'
number = '930 870 0299'

message = 'May kape kana, may tsismis kapa. Arat na!'

import datetime

now = datetime.datetime.now()
date_time = now.strftime("%Y-%m-%d %H:%M:%S")

receipt_complete = False

def login():
```

```
    global selected_option
    log = Toplevel()
    log.title("Tea Spillers")
    log.geometry("1355x650+100+140")
    log.resizable(width=False, height=False)
    frame = Frame(log, pady=250, padx=250)
    frame.pack()
    ##########################################################Header#######
##############################################
    headerLabel = Label(log, text="Tea Spillers",
                        font=("times new roman", 30, "bold"),
bg="RosyBrown", bd=12, relief=GROOVE)
    headerLabel.pack(fill=X)
    ##########################################################Customer
Details##############################################
    customer_Details_Frame = LabelFrame(log, text="Customer Details",
                                        font=("times new roman", 15,
"bold"), bg="RosyBrown", bd=12, relief=GROOVE)
    customer_Details_Frame.pack(fill=X)
    cstmr_name = Label(customer_Details_Frame, text="Nickname",
font=("times new roman", 15, "bold"), bg="RosyBrown")
    cstmr_name.grid(row=0, column=0, padx=20, pady=5)
    cstmr = "Enter a nickname"
    cstmr_entry = Entry(customer_Details_Frame, font=('arial', 15),
bd=7, width=18)
    cstmr_entry.insert(0, cstmr)
    cstmr_entry.grid(row=0, column=1, padx=8)
    cstmr_entry.bind('<Button-1>', lambda event: delete_zero(event,
cstmr_entry))
    phone_num = Label(customer_Details_Frame, text="Phone No.",
font=('times new roman', 15, 'bold'),
                      bg='RosyBrown')
    phone_num.grid(row=0, column=2, padx=20, pady=5)
    phone = "Enter a number"
    phone_num_entry = Entry(customer_Details_Frame, font=('arial', 15),
bd=7, width=18)
    phone_num_entry.insert(0, phone)
    phone_num_entry.grid(row=0, column=3, padx=8)
    phone_num_entry.bind('<Button-1>', lambda event: delete_zero(event,
phone_num_entry))
    bill_num = Label(customer_Details_Frame, text="Status Type",
font=('times new roman', 15, 'bold'),
                     bg='RosyBrown')
    bill_num.grid(row=0, column=5, padx=20, pady=5)
    selected_option = StringVar(customer_Details_Frame)
    # Create a list of options
    options = ["Student", "Regular", "N/A"]
    def check_option(*args):
```

```
        selected = selected_option.get()
        if selected not in options:
            messagebox.showerror("Invalid User Type", "Please select a
valid User Type.")
            return
    # Set the default selected option
    selected_option.trace('w', check_option)
    # Create the OptionMenu widget
    option_menu = OptionMenu(customer_Details_Frame, selected_option,
*options)
    option_menu.config(width=25)
    option_menu.grid(row=0, column=6)
    # Product Details
    prod_container = Frame(log)
    prod_container.pack()
    product_frame = LabelFrame(prod_container, text="Menu",
                               font=("times new roman", 18, "bold"),
bd=8, relief=GROOVE, bg='RosyBrown', padx=25)
    product_frame.grid(row=0,column=0, rowspan=1)
    def delete_zero(event, entry):
        entry.delete(0, 'end')

    origIcedCoffeePrice = 120.00
    origMatchaPrice = 145.00
    origAmericanoPrice = 160.00
    origCaffucinoPrice = 150.00

    cinnamonPrice = StringVar()
    churrosPrice = StringVar()
    muffinsPrice = StringVar()
    chocolatepiePrice = StringVar()

    cinnamonPrice.set(origIcedCoffeePrice)
    churrosPrice.set(origMatchaPrice)
    muffinsPrice.set(origAmericanoPrice)
    chocolatepiePrice.set(origCaffucinoPrice)

    ##################################################################
####RESET##########################################
    def reset():
        # Clear other fields and variables
        cstmr_entry.delete(0, END)
        matchaPrice.delete(0, END)
        textArea.config(state='normal')
        textArea.delete(1.0, END)
        textArea.config(state='disabled')
        americanoPrice.delete(0, END)
        caffucinoPrice.delete(0, END)
```

```
        cstmr_entry.insert(0, cstmr)
        ice.set(0)
        matcha.set(0)
        americano0.set(0)
        caffucino0.set(0)
        global receipt_complete
        receipt_complete = False
        receiptButton.configure(state=DISABLED)
        # Reset the add_to_cart() functionality
        global total_price, billing_complete
        total_price = 0
        billing_complete = False
        receipt_text_widget.config(state='normal')
        receipt_text_widget.delete('1.0', END)
        receipt_text_widget.config(state='disabled')
        CashAmt.place_forget()
        paymentBtn.place_forget()
        CashAmt.config(text="Cash:")
        textArea.config(state='normal')
        textArea.delete(1.0, END)
        textArea.config(state='disabled')
    ##################################################################Print
Page#########################################
    def print():
        global receipt_complete
        global cash_amount
        global payment_entry, selected_option
        # Check if the receipt is done
        if not receipt_complete:
            messagebox.showerror("Error", "Please complete the receipt
first.")
            return
        p = Toplevel()
        p.title("Tea Spillers | Your Receipt")
        p.geometry("1001x550+300+170")
        p.resizable(width=False, height=False)
        p.configure(bg="#fff")
        img = PhotoImage(file='logongteaspillers.png')
        Label(p, image=img, border=0, bg='white').place(x=5, y=15)
        p_frame = Frame(p, width=350, height=390, bg='white')
        p_frame.place(x=550, y=10)

        receiptHeader = Label(p_frame, text='Please check your folder
for the updated receipt copy.', bg='white', fg="salmon3",
font=('Helvetica', 12, 'bold'), wraplength=340)
        receiptHeader.place(x=130, y=10)

        prod_container = Frame(p, bg="RosyBrown")
```

```
        prod_container.place(x=540, y=70)

        receiptFrame = Frame(prod_container, bd=8, relief=GROOVE,
bg='RosyBrown')
        receiptFrame.grid(row=0, column=5, padx=10)

        receiptArea = Label(receiptFrame, text="Bill Area",
font=('times new roman', 15, 'bold'), bd=8, relief=GROOVE,
                            bg='RosyBrown')
        receiptArea.pack(fill=X)

        scrollbar = Scrollbar(receiptFrame, orient=VERTICAL)
        scrollbar.pack(side=RIGHT, fill=Y)
        textArea = Text(receiptFrame, width=50, height=25,
yscrollcommand=scrollbar.set)
        textArea.pack()

        scrollbar.config(command=textArea.yview)
        totalFrame = LabelFrame(root, font=("times new roman", 15,
"bold"), bg="RosyBrown", bd=15, relief=GROOVE)
        totalFrame.pack(fill=X)

        buyer_name = cstmr_entry.get()
        phone_numm = phone_num_entry.get()

        textArea.config(state='normal')
        textArea.delete(1.0, END)

        receipt_text =
'\t\t{}\n\t\t{}\n\t\t{}\n'.format(coffee_shop.title(),
shop_address.title(), number.title())
        textArea.insert(END, receipt_text)

        receipt_text1 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text1)

        receipt_text2 = f"{date_time[0:10]}\t\t\t\t{date_time[10:]}\n"
        textArea.insert(END, receipt_text2)

        if phone_numm.strip():
            receipt_phone = f"Customer no. {phone_numm}\n"
            textArea.insert(END, receipt_phone)
        else:
            receipt_phone1 = "Customer no.\n"
            textArea.insert(END, receipt_phone1)

        receipt_text3 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text3)
```

```
        receipt_text4 = '{}\n'.format(message.title())
        textArea.insert(END, receipt_text4)

        receipt_text5 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text5)

        receipt_text6 = "CAFE AND RESTAURANT\n\n"
        textArea.insert(END, receipt_text6)

        receipt_text7 = 'Quantity\t\tProduct Name\t\tProduct Price\n'
        textArea.insert(END, receipt_text7)

        # Retrieve the quantities from Entry widgets
        iced_coffee_quantity = int(ice.get())
        matcha_quantity = int(matcha.get())
        americano_quantity = int(americano0.get())
        caffucino_quantity = int(caffucino0.get())

        # Calculate the total price for each product
        iced_coffee_price = iced_coffee_quantity * origIcedCoffeePrice
        matcha_price = matcha_quantity * origMatchaPrice
        americano_price = americano_quantity * origAmericanoPrice
        caffucino_price = caffucino_quantity * origCaffucinoPrice

        # Include the quantities in the receipt if greater than zero
        if iced_coffee_quantity > 0:
            receipt_text8 = f'{iced_coffee_quantity}\t\tIced
Coffee\t\t{iced_coffee_price}\n'
            textArea.insert(END, receipt_text8)

        if matcha_quantity > 0:
            receipt_text9 =
f'{matcha_quantity}\t\tMatcha\t\t{matcha_price}\n'
            textArea.insert(END, receipt_text9)

        if americano_quantity > 0:
            receipt_text10 =
f'{americano_quantity}\t\tAmericano\t\t{americano_price}\n'
            textArea.insert(END, receipt_text10)

        if caffucino_quantity > 0:
            receipt_text11 =
f'{caffucino_quantity}\t\tCaffucino\t\t{caffucino_price}\n'
            textArea.insert(END, receipt_text11)

        receipt_text21 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text21)
```

```
        # Define the original prices as a dictionary
        original_prices = {
            'Iced Coffee': 120.00,
            'Matcha': 145.00,
            'Americano': 160.00,
            'Caffucino': 150.00
        }
        # Convert the quantity values to integers
        try:
            iced_coffee_quantity = int(iced_coffee_quantity)
            matcha_quantity = int(matcha_quantity)
            americano_quantity = int(americano_quantity)
            caffucino_quantity = int(caffucino_quantity)
        except ValueError:
            messagebox.showerror("Valid Quantity Error", "Please enter
a valid quantity for each drink.")
            return
        # Calculate the subtotals
        iced_coffee_subtotal = iced_coffee_quantity *
original_prices['Iced Coffee']
        matcha_subtotal = matcha_quantity * original_prices['Matcha']
        americano_subtotal = americano_quantity *
original_prices['Americano']
        caffucino_subtotal = caffucino_quantity *
original_prices['Caffucino']
        # Calculate the total price
        total_price = iced_coffee_subtotal + matcha_subtotal +
americano_subtotal + caffucino_subtotal
        # Calculate the discount based on the user type
        user_type = selected_option.get()
        discount = 0
        if user_type == "Student":
            discount = 0.05  # 5% discount for students
        elif user_type == "Regular":
            discount = 0.1  # 10% discount for regular users
    # Check if an option is selected
        if not selected_option.get():
            textArea.delete(1.0, END)  # Clear the textArea
            messagebox.showerror("Option Not Selected", "Please select
an option.")
            return
        # Apply the discount to the total price
        discounted_price = total_price - (total_price * discount)
        receipt_text12 = f'Subtotal:\t\t\t\t\t₱{total_price:.2f}\n'
        textArea.insert(END, receipt_text12)
        receipt_text13 = f'Discount: \t\t\t\t\t{discount * 100}%\n'
        textArea.insert(END, receipt_text13)
```

```
        receipt_text14 = f'Discounted Price:
\t\t\t\t\t₱{discounted_price:.2f}\n'
        textArea.insert(END, receipt_text14)
        receipt_text15 = ""
        payment_method = selected_option1.get()
        cash_amount = 0.0  # Default value
        receiptButton.config(state=DISABLED)

        if payment_method == "Cash" and payment_entry:
            cash_amount = payment_entry.get()
            try:
                cash_amount = float(cash_amount)
            except ValueError:
                messagebox.showerror("Valid Cash Amount Error", "Please
enter a valid cash amount.")

                return

            receipt_text15 = f'Cash:\t\t\t\t\t₱{cash_amount:.2f}\n'

        receipt_text15 += f"Payment Method: \t\t\t\t\t
{payment_method}\n"
        textArea.insert(END, receipt_text15)

        receipt_text16 = ""
        if payment_method == "Cash" and cash_amount >=
discounted_price:
            change = cash_amount - discounted_price
            receipt_text16 = f'Change:\t\t\t\t\t₱{change:.2f}\n'
            textArea.insert(END, receipt_text16)

        receipt_text17 = f'Total:\t\t\t\t\t₱{discounted_price:.2f}\n'
        textArea.insert(END, receipt_text17)
        receipt_text18 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text18)

        if buyer_name != cstmr:
            ending_message = f"Thank you for buying {buyer_name}!"
        else:
            ending_message = f"Thank you for your purchase!"
        ending_message_1 = " Find out about Tea Spillers at
teaspillers.com!"
        ending_message_text = '{}\t\t\t{}'.format(ending_message_1,
ending_message)

        receipt_text19 = f"{ending_message_text}\n"
        textArea.insert(END, receipt_text19)
```

```
        receipt_text20 = "=" * 50 + "\n"
        textArea.insert(END, receipt_text20)

        textArea.config(state='disabled')
        receipt_complete = True
        #Generating PDF
        pdf = FPDF()
        pdf.add_page()

        pdf.set_font("Arial", size=12)
        txt = (
            receipt_text +
            receipt_text1 +
            receipt_text2 +
            receipt_text3 +
            receipt_phone +
            receipt_text4 +
            receipt_text5 +
            receipt_text6 +
            receipt_text7 +
            receipt_text8 +
            receipt_text9 +
            receipt_text10 +
            receipt_text11 +
            receipt_text12 +
            receipt_text13 +
            receipt_text14 +
            receipt_text15 +
            receipt_text16 +
            receipt_text18 +
            receipt_text19 +
            receipt_text20 +
            receipt_text21
            )
        txt = textArea.get("1.0", "end")
        txt = txt.encode("latin-1", "replace").decode("latin-1")
        pdf.multi_cell(0,10, txt=txt)
        pdf.output("receipt.pdf")
        p.mainloop()
    ##################################################################
Receipt Page
    def receipt():
        check_option()
        global receipt_complete, billing_complete, selected_option1

        buyer_name = cstmr_entry.get()
        phone_numm = phone_num_entry.get()
```

```
        textArea.config(state='normal')
        textArea.delete(1.0, END)

        receipt_text =
'\t\t{}\n\t\t{}\n\t\t{}\n'.format(coffee_shop.title(),
shop_address.title(), number.title())
        textArea.insert(END, receipt_text)

        receipt_text1 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text1)

        receipt_text2 = f"{date_time[0:10]}\t\t\t\t{date_time[10:]}\n"
        textArea.insert(END, receipt_text2)

        if phone_numm.strip():
            receipt_phone = f"Customer no. {phone_numm}\n"
            textArea.insert(END, receipt_phone)
        else:
            receipt_phone1 = "Customer no.\n"
            textArea.insert(END, receipt_phone1)

        receipt_text3 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text3)

        receipt_text4 = '{}\n'.format(message.title())
        textArea.insert(END, receipt_text4)

        receipt_text5 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text5)

        receipt_text6 = "CAFE AND RESTAURANT\n\n"
        textArea.insert(END, receipt_text6)

        receipt_text7 = 'Quantity\t\tProduct Name\t\tProduct Price\n'
        textArea.insert(END, receipt_text7)

        # Retrieve the quantities from Entry widgets
        iced_coffee_quantity = int(ice.get())
        matcha_quantity = int(matcha.get())
        americano_quantity = int(americano0.get())
        caffucino_quantity = int(caffucino0.get())

        # Calculate the total price for each product
        iced_coffee_price = iced_coffee_quantity * origIcedCoffeePrice
        matcha_price = matcha_quantity * origMatchaPrice
        americano_price = americano_quantity * origAmericanoPrice
        caffucino_price = caffucino_quantity * origCaffucinoPrice
```

```
        # Include the quantities in the receipt if greater than zero
        if iced_coffee_quantity > 0:
            receipt_text8 = f'{iced_coffee_quantity}\t\tIced
Coffee\t\t{iced_coffee_price}\n'
            textArea.insert(END, receipt_text8)

        if matcha_quantity > 0:
            receipt_text9 =
f'{matcha_quantity}\t\tMatcha\t\t{matcha_price}\n'
            textArea.insert(END, receipt_text9)

        if americano_quantity > 0:
            receipt_text10 =
f'{americano_quantity}\t\tAmericano\t\t{americano_price}\n'
            textArea.insert(END, receipt_text10)

        if caffucino_quantity > 0:
            receipt_text11 =
f'{caffucino_quantity}\t\tCaffucino\t\t{caffucino_price}\n'
            textArea.insert(END, receipt_text11)

        receipt_text18 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text18)
        # Define the original prices as a dictionary
        original_prices = {
            'Iced Coffee': 120.00,
            'Matcha': 145.00,
            'Americano': 160.00,
            'Caffucino': 150.00
        }
        # Convert the quantity values to integers
        try:
            iced_coffee_quantity = int(iced_coffee_quantity)
            matcha_quantity = int(matcha_quantity)
            americano_quantity = int(americano_quantity)
            caffucino_quantity = int(caffucino_quantity)
        except ValueError:
            messagebox.showerror("Valid Quantity Error", "Please enter
a valid quantity for each drink.")
            return
        # Calculate the subtotals
        iced_coffee_subtotal = iced_coffee_quantity *
original_prices['Iced Coffee']
        matcha_subtotal = matcha_quantity * original_prices['Matcha']
        americano_subtotal = americano_quantity *
original_prices['Americano']
        caffucino_subtotal = caffucino_quantity *
original_prices['Caffucino']
```

```
        # Calculate the total price
        total_price = iced_coffee_subtotal + matcha_subtotal +
americano_subtotal + caffucino_subtotal
        # Calculate the discount based on the user type
        user_type = selected_option.get()
        discount = 0
        if user_type == "Student":
            discount = 0.05  # 5% discount for students
        elif user_type == "Regular":
            discount = 0.1  # 10% discount for regular users

        if not selected_option.get():
            textArea.delete(1.0, END)  # Clear the textArea
            messagebox.showerror("Option Not Selected", "Please select
an option.")
            return

        # Apply the discount to the total price
        discounted_price = total_price - (total_price * discount)
        receipt_text12 = f'Subtotal:\t\t\t\t\t\t₱{total_price:.2f}\n'
        textArea.insert(END, receipt_text12)
        receipt_text13 = f'Discount: \t\t\t\t\t{discount * 100}%\n'
        textArea.insert(END, receipt_text13)

        receipt_text14 = f'Discounted Price:
\t\t\t\t\t₱{discounted_price:.2f}\n'
        textArea.insert(END, receipt_text14)

        receipt_text15 = ""
        payment_method = selected_option1.get()
        cash_amount = 0.0  # Default value

        if payment_method == "Cash" and payment_entry:
            cash_amount = payment_entry.get()
            try:
                cash_amount = float(cash_amount)
            except ValueError:
                messagebox.showerror("Valid Cash Amount Error", "Please
enter a valid cash amount.")
                receiptButton.configure(state=NORMAL)
                return
            receipt_text15 = f'Cash:\t\t\t\t\t\t₱{cash_amount:.2f}\n'

        receipt_text15 += f"Payment Method:
      \t\t\t\t\t{payment_method}\n"
        textArea.insert(END, receipt_text15)
```

```
        if payment_method == "Cash" and cash_amount >=
discounted_price:
            change = cash_amount - discounted_price
            receipt_text16 = f'Change:\t\t\t\t\t₱{change:.2f}\n'
            textArea.insert(END, receipt_text16)

        receipt_text17 = f'Total:\t\t\t\t\t₱{discounted_price:.2f}\n'
        textArea.insert(END, receipt_text17)

        receipt_text18 = "-" * 50 + "\n"
        textArea.insert(END, receipt_text18)

        if buyer_name != cstmr:
            ending_message = f"Thank you for buying {buyer_name}!"
        else:
            ending_message = f"Thank you for your purchase!"
        ending_message_1 = " Find out about Tea Spillers at
teaspillers.com!"
        ending_message_text = '{}\t\t\t{}'.format(ending_message_1,
ending_message)

        receipt_text19 = f"{ending_message_text}\n"
        textArea.insert(END, receipt_text19)

        receipt_text20 = "=" * 50 + "\n"
        textArea.insert(END, receipt_text20)

        textArea.config(state='disabled')
        receipt_complete = True
    # Desserts
    quantity1 = 0
    quantity2 = 0
    quantity3 = 0
    quantity4 = 0

    ice = StringVar()
    ice.set(str(quantity1))
    matcha = StringVar()
    matcha.set(str(quantity2))
    americano0 = StringVar()
    americano0.set(str(quantity3))
    caffucino0 = StringVar()
    caffucino0.set(str(quantity4))
      # Set an initial value for the StringVar
    def increment_quantity(quantity_var):
        current_quantity = int(quantity_var.get())
        new_quantity = current_quantity + 1
        quantity_var.set(str(new_quantity))
```

```
    def decrement_quantity(quantity_var1):
        current_quantity = int(quantity_var1.get())
        if current_quantity > 0:
            new_quantity = current_quantity - 1
            quantity_var1.set(str(new_quantity))
    def reset_quantities():
        global quantity1, quantity2, quantity3, quantity4
        global option_label, payment_entry

        quantity1 = 0
        ice.set(str(quantity1))
        quantity2 = 0
        matcha.set(str(quantity2))
        quantity3 = 0
        americano0.set(str(quantity3))
        quantity4 = 0
        caffucino0.set(str(quantity4))
    # Clear the receipt text
        receipt_text_widget.config(state="normal")  # Set the state to
normal to enable modifications
        receipt_text_widget.delete("1.0", END)  # Clear the text area
        receipt_text_widget.config(state="disabled")  # Set the state
to disabled to make it read-only

        paymentBtn.place_forget()
        CashAmt.place_forget()

        CashAmt.config(text="Cash:")

    def create_options():
        global payment_entry, cash, billing_complete
        cash = Toplevel()
        cash.title("Tea Spillers | Payment Method")
        cash.geometry("900x500+300+200")
        cash.configure(bg="#fff")
        cash.resizable(width=False, height=False)

        img_path = 'logongteaspillers.png'
        image = Image.open(img_path)
        imguli = ImageTk.PhotoImage(image)

        label = Label(cash, image=imguli, bg='white')
        label.image = imguli  # Save a reference to the image to
prevent it from being garbage collected
        label.place(x=10, y=10)

        receiptButton.config(state=DISABLED)
```

```
        cash_frame = Frame(cash, width=350, height=420, bg='white')
        cash_frame.place(x=550, y=50)

        payHeader = Label(cash_frame, text='Payment Method',
fg="salmon3", bg='white', font=('Helvetica', 23, 'bold'))
        payHeader.place(x=70, y=20)
        totalis = Label(cash_frame, text=f"Your Total Price is :
{total_price}", bg='white').place(x=40,y=120)
        global option_label, payment_entry, selected_option1
        selected_option1 = StringVar(cash_frame)

    # Create a list of options
        option_label = Label(cash_frame, text="Type of
Payment:",bg="white").place(x=40,y=155)
        options = ["Cash", "E-Wallet(GCash)"]

    # Set the default selected option
        selected_option1.set(options[0])
        def option_selected(*args):
            global payment_entry, gcash_entry
            selected = selected_option1.get()
            if selected == "Cash":
                textt = "Enter payment Amount..."
                payment_entry = Entry(cash_frame, font=("Arial", 12),
width=25)
                payment_entry.insert(0, textt)
                payment_entry.place(x=50, y=220)
                doneBtn = Button(cash_frame, text="Done", bg="salmon3",
command=done_clicked).place(x=70, y=250)

                payment_entry.bind('<Button-1>', lambda event:
delete_zero(event, payment_entry))
            elif selected == "E-Wallet(GCash)":
                textt = "Enter 11-digit number..."
                gcash_entry = Entry(cash_frame, font=("Arial", 12),
width=25)  # Use gcash_entry instead of payment_entry
                gcash_entry.insert(0, textt)
                gcash_entry.place(x=50, y=220)
                doneBtn = Button(cash_frame, text="Done", bg="salmon3",
command=done_clicked).place(x=70, y=250)

                def delete_text(event):
                    gcash_entry.delete(0, END)

                gcash_entry.bind('<Button-1>', delete_text)
                gcash_entry.bind('<FocusOut>', validate_gcash_number)

        def validate_gcash_number(event):
```

```
            gcash_number = gcash_entry.get()  # Use gcash_entry instead
of payment_entry
            if len(gcash_number) != 11 or not gcash_number.isdigit():
                messagebox.showerror("Invalid GCash Number", "Please
enter a valid 11-digit GCash number.")
                gcash_entry.delete(0, END)
                gcash_entry.insert(0, "Enter 11-digit number...")
                cash_frame.focus_force()
                CashAmt.config(text="")  # Clear the CashAmt label
            else:
                CashAmt.config(text="Paid by GCash")  # Update the
CashAmt label

        def done_clicked():
            receiptButton.config(state=DISABLED)
            global cash_amt, total_price, payment_entry

            if selected_option1.get() == "Cash":
                cash_amt = payment_entry.get()
            elif selected_option1.get() == "E-Wallet(GCash)":
                cash_amt = gcash_entry.get()

            CashAmt.config(text=f"Cash: {cash_amt}")

            if CashAmt.cget("text") == "Cash: ":  # Check if CashAmt
label text is empty
                receiptButton.config(state=DISABLED)  # Disable the
receipt button
            else:
                try:
                    cash_amt = float(cash_amt)
                    if cash_amt >= total_price:
                        # Payment is sufficient, proceed with further
actions
                        cash.withdraw()  # Hide the cash window
                    else:
                        # Insufficient payment, display error message
                        messagebox.showerror("Insufficient Payment",
"The payment amount is insufficient.")
                        if selected_option1.get() == "Cash":
                            payment_entry.delete(0, 'end')  # Reset the
payment entry field
                            payment_entry.insert(0, "Enter payment
amount...")
                        elif selected_option1.get() == "E-
Wallet(GCash)":
                            gcash_entry.delete(0, 'end')  # Reset the
GCash entry field
```

```
                            gcash_entry.insert(0, "Enter 11-digit
number...")
                        CashAmt.config(text="")  # Clear the CashAmt
label

                except ValueError:
                    # Invalid payment amount, display error message
                    messagebox.showerror("Invalid Payment", "Please
enter a valid payment amount.")
                    if selected_option1.get() == "Cash":
                        payment_entry.delete(0, 'end')  # Reset the
payment entry field
                        payment_entry.insert(0, "Enter payment
amount...")
                    elif selected_option1.get() == "E-Wallet(GCash)":
                        gcash_entry.delete(0, 'end')  # Reset the GCash
entry field
                        gcash_entry.insert(0, "Enter 11-digit
number...")
                    CashAmt.config(text="")  # Clear the CashAmt label

            receiptButton.config(state=NORMAL)  # Enable the receipt
button

    # Bind the option_selected function to the OptionMenu widget
        selected_option1.trace("w", option_selected)

    # Create the OptionMenu widget
        option_menu = OptionMenu(cash_frame, selected_option1,
*options)
        option_menu.config(width=20)

    # Place the OptionMenu widget below the receipt text widget
        option_menu.place(x=150,y=150)
        billing_complete = False

        #TotalFrame
    coffee_frame = Frame(prod_container)
    coffee_frame.grid(row=0, column=1, sticky="ns")

    drinks_frame = LabelFrame(coffee_frame, text="Billing Area",
font=("times new roman", 18, "bold"), bd=8,
                              relief=GROOVE, bg='RosyBrown', padx=0,
width=280, height=200)
    drinks_frame.grid()

    receipt_text7 = 'Quantity\tProduct Name\t\tProduct Price\n'
```

```
    receipt_text_widget = Text(drinks_frame, font=("Times New Roman",
12), bg="RosyBrown", width=38, height=18)
    receipt_text_widget.insert("1.0", receipt_text7)
    paymentBtn = Button(drinks_frame, text="Click here to select
payment method", bg="salmon3",command=create_options)
    CashAmt = Label(drinks_frame, text="Cash:",font=("Times New Roman",
12), bg="RosyBrown")

    receipt_text_widget.grid(padx=5, pady=10)

    original_prices = {
        'Iced Coffee': 120.00,
        'Matcha': 145.00,
        'Americano': 160.00,
        'Caffucino': 150.00
    }

    def add_to_cart():

        global total_price, billing_complete
        quantities = [int(ice.get()), int(matcha.get()),
int(americano0.get()), int(caffucino0.get())]
        products = ["Iced Coffee", "Matcha", "Americano", "Caffucino"]

        receipt_text = "Quantity\tProduct Name\t\tProduct Price\n"
        global total_price
        total_price = 0

        for i in range(len(quantities)):
            if quantities[i] > 0:
                product_name = products[i]
                quantity = quantities[i]
                product_price = original_prices[product_name]
                line_total = product_price * quantity

                receipt_text +=
f"{quantity}\t{product_name}\t\t{line_total:.2f}\n"
                total_price += line_total

        receipt_text += f"\nTotal Price: {total_price:.2f}"
        CashAmt.place(x=8,y=150)
        paymentBtn.place(x=70,y=180)
        billing_complete = TRUE
        receipt_text_widget.config(state="normal")
        receipt_text_widget.delete("1.0", END)
        receipt_text_widget.insert(END, receipt_text)
        receipt_text_widget.config(state="disabled")
```

```
#Menu
    cinnamonLabel = Button(product_frame,
                           bg="beige", borderwidth=10,command=lambda:
increment_quantity(ice))

    icedImg = PhotoImage(file="match.png")
    res = icedImg.subsample(2)

    cinnamonLabel.config(image=res, compound=LEFT)
    cinnamonLabel.config(width=70,height=50)
    cinnamonLabel.grid(row=1, column=0, sticky="w", padx=20)

    icedLabel = Label(product_frame, text="Iced Coffee", font=("times
new roman", 15, "bold"),
                          bg="RosyBrown")
    icedLabel.grid(row=1, column=1, sticky="w",padx=20)

    prodLabel = Label(product_frame, text="Products", font=('arial',
15, 'bold'), bg='RosyBrown').grid(row=0, column=1)
    priceLabel = Label(product_frame, text="Price", font=('arial', 15,
'bold'), bg='RosyBrown').grid(row=0,column=2)
    quantLabel = Label(product_frame, text="Quantity", font=('arial',
15, 'bold'), bg='RosyBrown').grid(row=0, column=3)
    cinnamonPriceEntry = Entry(product_frame, font=("arial", 15,),
bd=10, width=5, justify='center',
                               textvariable=cinnamonPrice,
state='readonly')
    cinnamonPriceEntry.grid(row=1, column=2, padx=10)


    icedCoffeePrice = Entry(product_frame, font=("arial", 15), bd=10,
width=3, justify='center', textvariable=ice)
    icedCoffeePrice.grid(row=1, column=3, padx=8, pady=8)
    icedCoffeePrice.bind('<Button-1>', lambda event: delete_zero(event,
icedCoffeePrice))

    minusBtn = Button(product_frame,text="-", bd=10, width=3,
justify='center', bg="tomato3",command=lambda:
decrement_quantity(ice)).grid(row=1, column=4)

    matchaLabel = Button(product_frame, font=("times new roman", 15,
"bold"),
                          bg="beige", borderwidth=10,command=lambda:
increment_quantity(matcha))
    matchaLabel.grid(row=2, column=0, sticky="w", padx=20)

    matchImg = PhotoImage(file="glass.png")
    res1 = matchImg.subsample(2)
```

```
    matchaLabel.config(image=res1, compound=LEFT)
    matchaLabel.config(width=70, height=50)

    matchaaLabel = Label(product_frame, text="Matcha", font=("times new
roman", 15, "bold"),
                         bg="RosyBrown")
    matchaaLabel.grid(row=2, column=1, sticky="w", padx=20)

    matchaPriceEntry = Entry(product_frame, font=("arial", 15), bd=10,
width=5, justify='center',
                             textvariable=churrosPrice,
state='readonly')

    matchaPriceEntry.grid(row=2, column=2, padx=10, pady=10)

    matchaPrice = Entry(product_frame, font=("arial", 15), bd=10,
width=3, justify='center', textvariable=matcha)
    matchaPrice.grid(row=2, column=3, padx=8, pady=8)
    matchaPrice.bind('<Button-1>', lambda event: delete_zero(event,
matchaPrice))

    minusBtn = Button(product_frame, text="−", bd=10, width=3,
justify='center', bg="tomato3",command=lambda:
decrement_quantity(matcha)).grid(row=2, column=4)

    AmericanoLabel = Button(product_frame, font=("times new roman", 15,
"bold"),
                         bg="beige", borderwidth=10,command=lambda:
increment_quantity(americano0))
    AmericanoLabel.grid(row=3, column=0, sticky="w", padx=20)

    ameImg = PhotoImage(file="americano.png")
    res2 = ameImg.subsample(2)

    AmericanoLabel.config(image=res2, compound=LEFT)
    AmericanoLabel.config(width=70, height=50)

    AmeLabel = Label(product_frame, text="Americano", font=("times new
roman", 15, "bold"),
                         bg="RosyBrown")
    AmeLabel.grid(row=3, column=1, sticky="w", padx=20)
    muffinsPriceEntry = Entry(product_frame, font=("arial", 15), bd=10,
width=5, justify='center',
                             textvariable=muffinsPrice,
state='readonly')
    muffinsPriceEntry.grid(row=3, column=2, padx=10, pady=10)
```

```
    americanoPrice = Entry(product_frame, font=("arial", 15), bd=10,
width=3, justify='center', textvariable=americano0)
    americanoPrice.grid(row=3, column=3, padx=8, pady=8)
    americanoPrice.bind('<Button-1>', lambda event: delete_zero(event,
americanoPrice))

    minusBtn = Button(product_frame, text="–", bd=10, width=3,
justify='center', bg="tomato3",command=lambda:
decrement_quantity(americano0)).grid(row=3, column=4)

    chocolatepieLabel = Button(product_frame, font=("times new roman",
15, "bold"),
                               bg="beige",
borderwidth=10,command=lambda: increment_quantity(caffucino0))
    chocolatepieLabel.grid(row=4, column=0, sticky="w", padx=20)

    cafImg = PhotoImage(file="coffee.png")
    res3 = cafImg.subsample(2)

    chocolatepieLabel.config(image=res3, compound=LEFT)
    chocolatepieLabel.config(width=70, height=50)
    CafLabel = Label(product_frame, text="Caffucino", font=("times new
roman", 15, "bold"),
                         bg="RosyBrown")
    CafLabel.grid(row=4, column=1, sticky="w", padx=20)
    chocolatepiePriceEntry = Entry(product_frame, font=("arial", 15),
bd=10, width=5, justify='center',
                                   textvariable=chocolatepiePrice,
state='readonly')
    chocolatepiePriceEntry.grid(row=4, column=2, padx=10, pady=10)

    caffucinoPrice = Entry(product_frame, font=("arial", 15), bd=10,
width=3, justify='center', textvariable=caffucino0)
    caffucinoPrice.grid(row=4, column=3, padx=8, pady=8)

    caffucinoPrice.bind('<Button-1>', lambda event: delete_zero(event,
caffucinoPrice))
    minusBtn = Button(product_frame, text="–", bd=10, width=3,
justify='center', bg="tomato3",command=lambda:
decrement_quantity(caffucino0)).grid(row=4, column=4)

    RemoveLabel = Button(product_frame, font=("times new roman", 15,
"bold"),width=5,padx=5, pady=5,
bg="RosyBrown",command=reset_quantities, borderwidth=0)
    binImg = PhotoImage(file="bin.png")
    res5 = binImg.subsample(22)

    RemoveLabel.config(image=res5, compound=LEFT)
```

```
    RemoveLabel.config(width=50,height=50)

    RemoveLabel.grid(row=5, column=4, sticky="w")

    addBtn = Button(product_frame, text="Add", font=("times new roman",
15, "bold"),width=5,padx=5, pady=5, bg="beige",
command=add_to_cart).grid(row=5, column=3, sticky="w")

    ##########################################################Receipt

    receiptFrame = Frame(prod_container, bd=8, relief=GROOVE)
    receiptFrame.grid(row=0, column=2, padx=10, sticky='n')

    receiptArea = Label(receiptFrame, text="Click Receipt to view your
Bill", font=('times new roman', 15, 'bold'),
                        bd=8, relief=GROOVE)
    receiptArea.pack(fill=X)

    scrollbar = Scrollbar(receiptFrame, orient=VERTICAL)
    scrollbar.pack(side=RIGHT, fill=Y)

    textArea = Text(receiptFrame, width=50, height=20,
yscrollcommand=scrollbar.set)
    textArea.pack()

    scrollbar.config(command=textArea.yview)

    totalFrame = LabelFrame(log, font=("times new roman", 15, "bold"),
bg="RosyBrown", bd=15, relief=GROOVE, height=50)
    totalFrame.pack(fill=X)

    ##############################################################EXIT
    def Exit():
        result = messagebox.askyesnocancel("Billing System", "Do you
want to order again?")
        if result is None:
            return  # User clicked on the 'Cancel' button, do nothing
        elif result:
            log.withdraw()  # Destroy the current window
            log.deiconify()  # Show the login window again
            reset()
            entry1.config(show="")
        else:
            log.withdraw()
            root.deiconify()
            reset()
            entry1.config(show="")
```

```
    totalButton = Button(totalFrame, text="Total", font=("times new
roman", 15, "bold"), bg="beige", fg="black",
                         width=10, padx=5, pady=5, command=add_to_cart)
    totalButton.grid(row=0, column=0, padx=80, pady=10)

    billing_complete = False
    receiptButton = Button(totalFrame, text="Receipt", font=("times new
roman", 15, "bold"), bg="beige", fg="black",
                           width=10, padx=5, pady=5, command=receipt,
state=DISABLED)
    receiptButton.grid(row=0, column=1, padx=50, pady=10)

    printButton = Button(totalFrame, text="Print", font=("times new
roman", 15, "bold"), bg="beige", fg="black",
                         width=10, padx=5, pady=5, command=print)
    printButton.grid(row=0, column=2, padx=50, pady=10)

    resetButton = Button(totalFrame, text="Reset", command=reset,
font=("times new roman", 15, "bold"), bg="beige",
                         fg="black", width=10, padx=5, pady=5)
    resetButton.grid(row=0, column=3, padx=50, pady=10)

    exitButton = Button(totalFrame, text="Exit", command=Exit,
font=("times new roman", 15, "bold"), bg="beige",
                        fg="black", width=10, padx=5, pady=5)
    exitButton.grid(row=0, column=4, padx=50, pady=10)
    log.mainloop()
################################################################ Log in
# Creating Label, Entries, and button
user_accounts = {
    "user": "Admin12345",
    "1": "2"
}

def reset():
    entry.delete(0, END)
    entry.insert(0, "Username")  # Set the default username value
    entry1.delete(0, END)
    entry1.insert(0, "Password")  # Set the default password value

def signin():
    entered_username = entry.get()
    entered_password = entry1.get()

    if entered_username in user_accounts and
user_accounts[entered_username] == entered_password:
        messagebox.showinfo("Login", "You have successfully log in!")
        root.withdraw()
```

```
        reset()
        login()
    else:
        messagebox.showerror("Login", "Invalid username or password.")
        reset()

# LOGIN
def clear_username(event):
    entry.delete(0, END)

def clear_password(event):
    entry1.delete(0, END)
    entry1.config(show="*")

frame = Frame(root, width=350, height=350, bg='white')
frame.place(x=550, y=70)

signinHeader = Label(frame, text='Welcome Back', fg="salmon3",
bg='white', font=('Helvetica', 23, 'bold'))
signinHeader.place(x=70, y=5)

entry = Entry(frame, width=25, fg='black', border=0, bg='white',
font=('Helvetica', 11))
entry.place(x=30, y=100)
entry.insert(0, 'Username')
entry.bind("<Button-1>", clear_username)

Frame(frame, width=295, height=2, bg='black').place(x=25, y=127)

entry1 = Entry(frame, width=25, fg='black', border=0, bg='white',
font=('Helvetica', 11))
entry1.place(x=30, y=177)
entry1.insert(0, 'Password')
entry1.bind("<Button-1>", clear_password)

Frame(frame, width=295, height=2, bg='black').place(x=25, y=207)

SigninBtn = Button(frame, width=15, pady=7, text='Sign in',
bg='salmon3', fg='white', border=0, command=signin).place(
    x=55, y=274)
#################################################################SignU
P
def signupp():
    root.withdraw()
    signup = Toplevel()
    signup.title("Sign up | Tea Spillers")
    signup.geometry("951x550+300+200")
    signup.resizable(width=False, height=False)
```

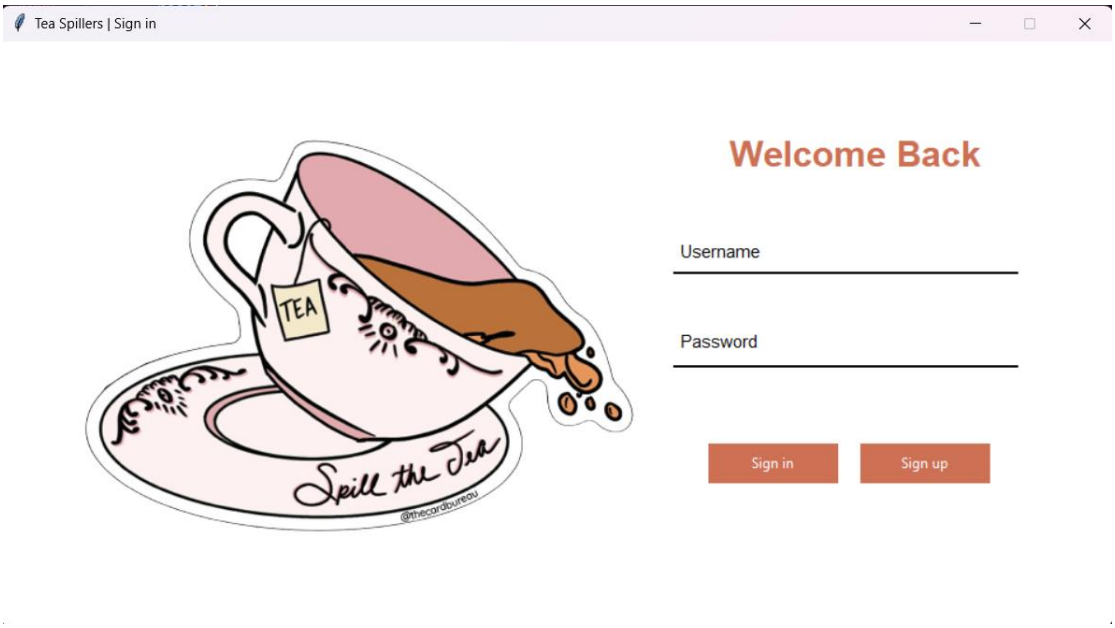
```
    signup.configure(bg="#fff")

    def create_account():
        entered_username = user.get()
        entered_password = password.get()
        entered_email = email.get()
        entered_confirm_password = password1.get()

        if entered_username == "" or entered_password == "" or
entered_email == "" or entered_confirm_password == "":
            messagebox.showerror("Registration", "Please fill in all
the fields.")
        elif entered_password != entered_confirm_password:
            messagebox.showerror("Registration", "Passwords do not
match.")
            password.config(show="")
            password1.config(show="")
            reset1()
            signup.deiconify()
        elif not is_password_strong(entered_password):
            messagebox.showerror("Registration", "Password is not
strong enough.")
            password.config(show="")
            password1.config(show="")
            reset1()
        elif entered_username in user_accounts:
            messagebox.showerror("Registration", "Account already
exists with this username.")
            password1.config(show="")
            reset1()
        else:
            user_accounts[entered_username] = entered_password
            messagebox.showinfo("Registration", "Account created
successfully! You can now log in.")
            signup.withdraw()
            root.deiconify()

    def is_password_strong(password):
        # Check if password meets the required strength criteria
        # At least 8 characters long, contains at least one uppercase
letter, one lowercase letter, and one digit
        if len(password) < 8:
            return False
        has_lowercase = False
        has_uppercase = False
        has_digit = False
        for char in password:
            if char.islower():
```

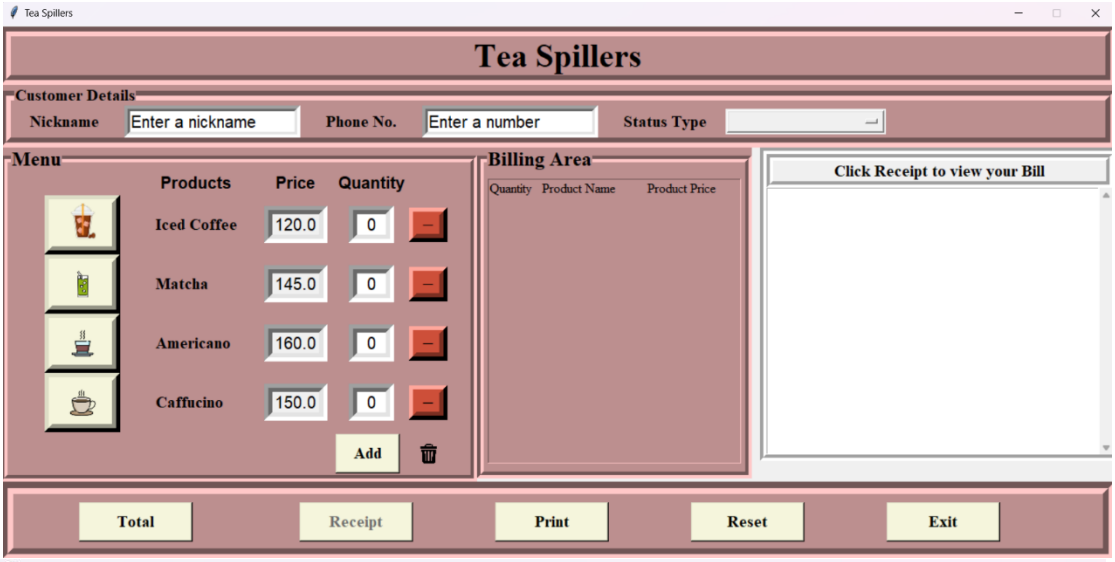
```
                has_lowercase = True
            elif char.isupper():
                has_uppercase = True
            elif char.isdigit():
                has_digit = True
        return has_lowercase and has_uppercase and has_digit

    img_signup = PhotoImage(file='logongteaspillers.png')
    Label(signup, image=img_signup, border=0, bg='white').place(x=20,
y=20)

    signup_frame = Frame(signup, width=350, height=420, bg='white')
    signup_frame.place(x=550, y=50)

    signinHeader = Label(signup_frame, text='Create Account',
fg="salmon3", bg='white', font=('Helvetica', 23, 'bold'))
    signinHeader.place(x=70, y=20)

    def clear_username1(event):
        user.delete(0, END)

    def clear_email1(event):
        email.delete(0, END)

    def clear_password1(event):
        password.delete(0, END)
        password.config(show="*")

    def clear_confirmpassword1(event):
        password1.delete(0, END)
        password1.config(show="*")

    def reset1():
        user.delete(0, END)
        user.insert(0, "Username")
        email.delete(0, END)
        email.insert(0, "Email")
        password.delete(0, END)
        password.insert(0, "Password")
        password1.delete(0, END)
        password1.insert(0, "Confirm Password")

    user = Entry(signup_frame, width=25, fg='black', border=0,
bg='white', font=('Helvetica', 11))
    user.place(x=30, y=80)
    user.insert(0, 'Username')
    user.bind("<Button-1>", clear_username1)
```

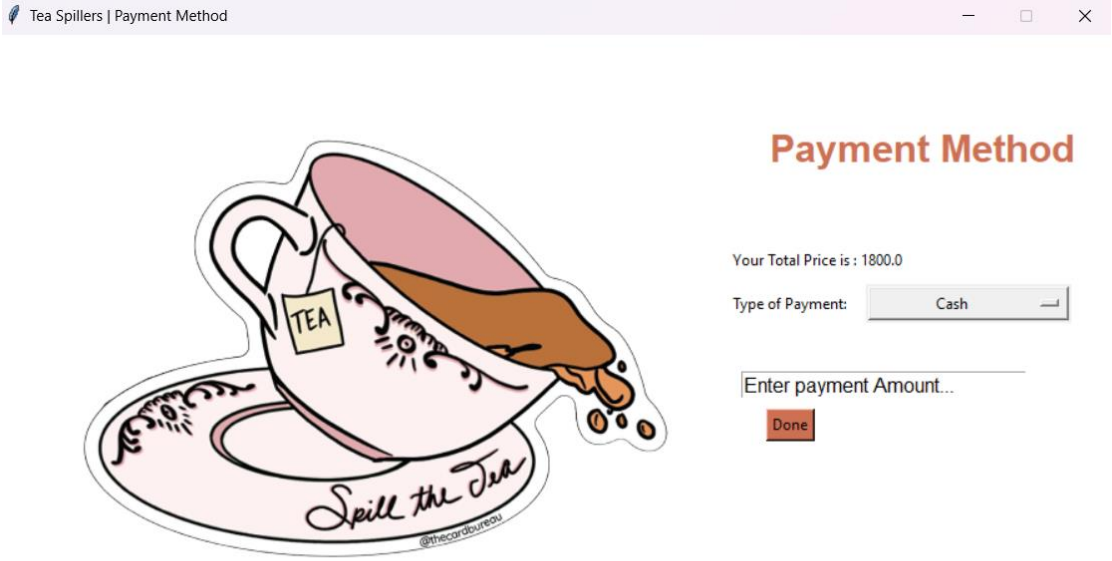
```
    Frame(signup_frame, width=295, height=2, bg='black').place(x=25,
y=117)

    email = Entry(signup_frame, width=25, fg='black', border=0,
bg='white', font=('Helvetica', 11))
    email.place(x=30, y=150)
    email.insert(0, 'Email')
    email.bind("<Button-1>", clear_email1)

    Frame(signup_frame, width=295, height=2, bg='black').place(x=25,
y=187)

    password = Entry(signup_frame, width=25, fg='black', border=0,
bg='white', font=('Helvetica', 11))
    password.place(x=30, y=220)
    password.insert(0, 'Password')
    password.bind("<Button-1>", clear_password1)

    Frame(signup_frame, width=295, height=2, bg='black').place(x=25,
y=257)

    password1 = Entry(signup_frame, width=25, fg='black', border=0,
bg='white', font=('Helvetica', 11))
    password1.place(x=30, y=290)
    password1.insert(0, 'Confirm Password')
    password1.bind("<Button-1>", clear_confirmpassword1)

    Frame(signup_frame, width=295, height=2, bg='black').place(x=25,
y=327)

    Label(signup_frame, text="The password should be at least 8
characters, 1 uppercase, 1 lowercase, and 1 digit.", font=('Arial', 8),
bg="white", wraplength=305).place(x=25, y=330)

    SubmitBtn = Button(signup_frame, width=15, pady=7, text='Submit',
bg='salmon3', fg='white', border=0,
                       command=create_account)
    SubmitBtn.place(x=45, y=389)

    def cancel():
        signup.withdraw()
        reset()
        root.deiconify()
        reset()
        entry1.config(show="")

    CancelBtn = Button(signup_frame, width=15, pady=7, text='Cancel',
bg='salmon3', fg='white', border=0,
```

```
                        command=cancel)
    CancelBtn.place(x=175, y=389)
    signup.mainloop()

SignupBtn = Button(frame, width=15, pady=7, text='Sign up',
bg='salmon3', fg='white', border=0, command=signupp).place(
    x=185, y=274)

root.mainloop()
```

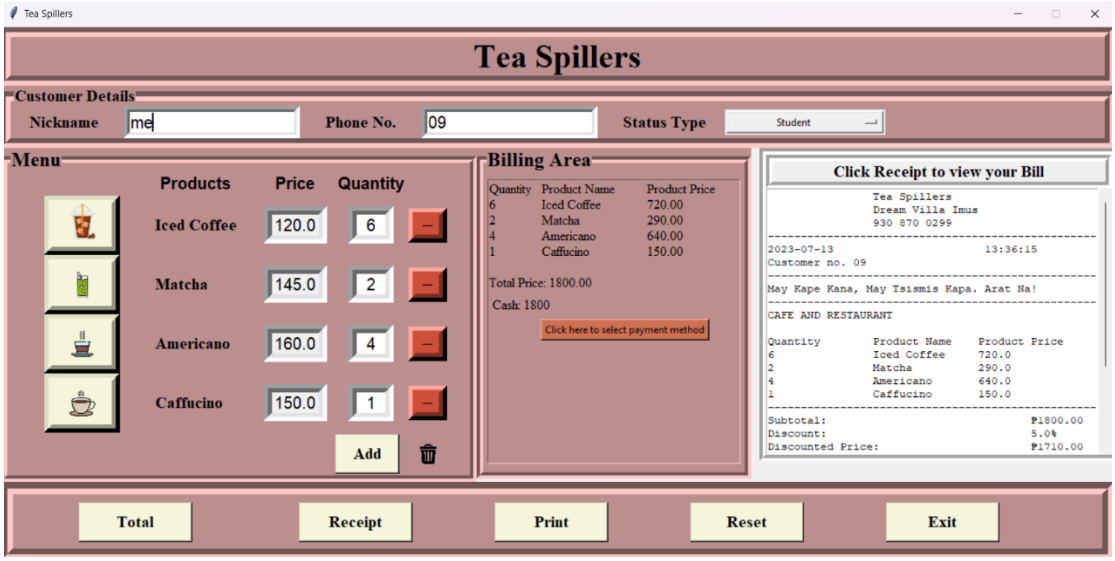
System Output

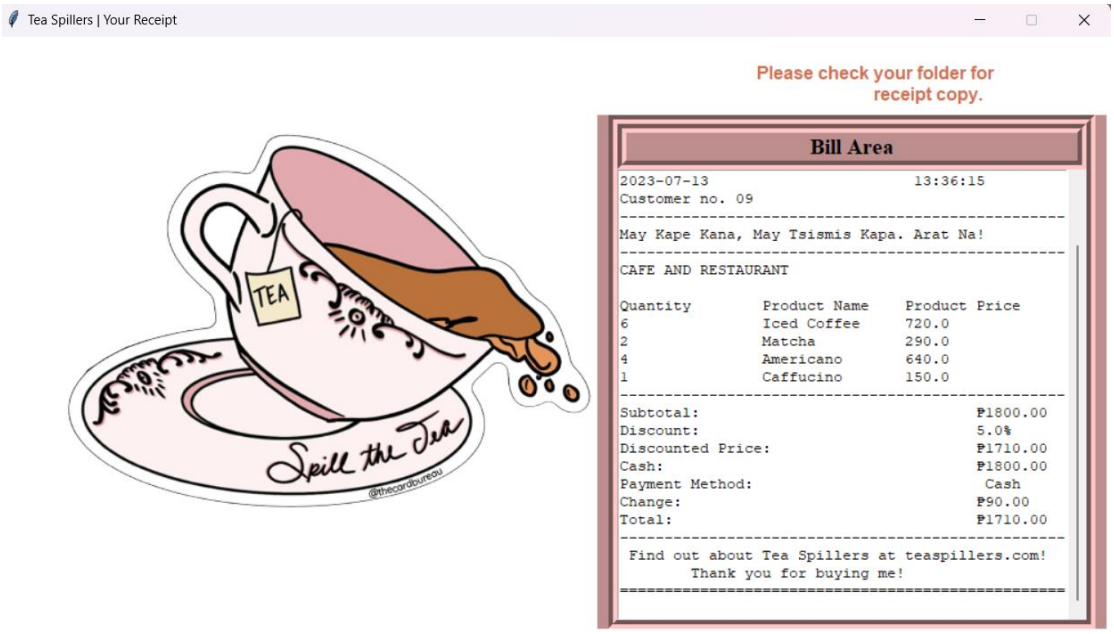



Tea Spillers
Tea Spillers
Customer Details
Nickname
Enter a nickname
Phone No.
Enter a number
Status Type
Menu
Products
Price
Quantity
Iced Coffee
120.0
0
Matcha
145.0
0
Americano
160.0
0
Caffucino
150.0
0
Add
Billing Area
Quantity
Product Name
Product Price
Click Receipt to view your Bill
Total
Receipt
Print
Reset
Exit


Tea Spillers | Payment Method
TEA
Spill the Tea
@thecardoureau
Payment Method
Your Total Price is : 1800.0
Type of Payment:
Cash
Enter payment Amount...
Done


Tea Spillers | Payment Method
TEA
Spill the Tea
@thecardoureau
Payment Method
Your Total Price is : 1800.0
Type of Payment:
E-Wallet(GCash)
Enter 11-digit number...
Done


Tea Spillers
Tea Spillers
Customer Details
Nickname
me
Phone No.
09
Status Type
Student
Menu
Products
Price
Quantity
Iced Coffee
120.0
6
Matcha
145.0
2
Americano
160.0
4
Caffucino
150.0
1
Add
Billing Area
Quantity Product Name Product Price
6 Iced Coffee 720.00
2 Matcha 290.00
4 Americano 640.00
1 Caffucino 150.00
Total Price: 1800.00
Cash: 1800
Click here to select payment method
Click Receipt to view your Bill
Tea Spillers
Dream Villa Imus
930 870 0299
2023-07-13 13:36:15
Customer no. 09
May Kape Kana, May Tsismis Kapa. Arat Na!
CAFE AND RESTAURANT
Quantity Product Name Product Price
6 Iced Coffee 720.0
2 Matcha 290.0
4 Americano 640.0
1 Caffucino 150.0
Subtotal: ₱1800.00
Discount: 5.0%
Discounted Price: ₱1710.00
Total
Receipt
Print
Reset
Exit


Tea Spillers | Your Receipt
Please check your folder for receipt copy.
TEA
Spill the Tea
@thecardourseu
Bill Area
2023-07-13 13:36:15
Customer no. 09
May Kape Kana, May Tsismis Kapa. Arat Na!
CAFE AND RESTAURANT
Quantity Product Name Product Price
6 Iced Coffee 720.0
2 Matcha 290.0
4 Americano 640.0
1 Caffucino 150.0
Subtotal: ₱1800.00
Discount: 5.0%
Discounted Price: ₱1710.00
Cash: ₱1800.00
Payment Method: Cash
Change: ₱90.00
Total: ₱1710.00
Find out about Tea Spillers at teaspillers.com!
Thank you for buying me!