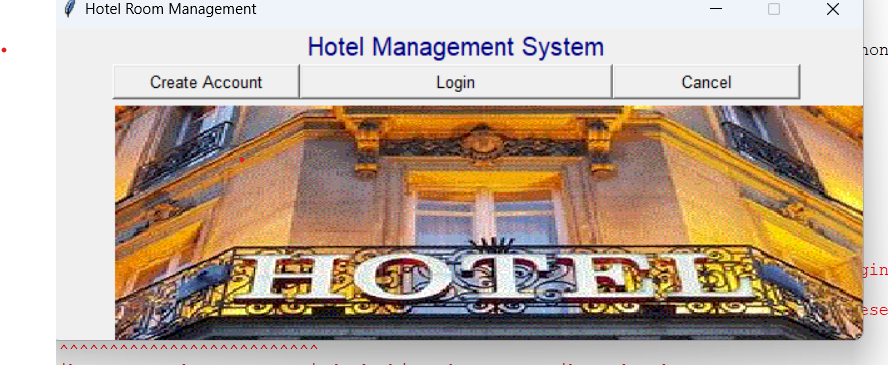
Asma Naweed

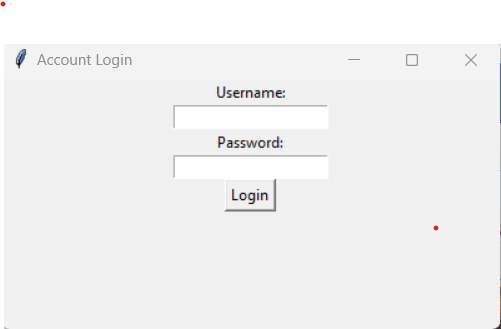
CSE-222/ Hotel Management Project

Milestone#2

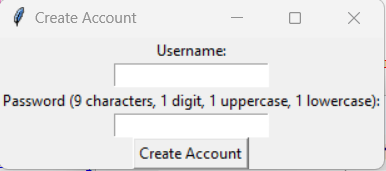
**Screen Captures of the Program:**

Main Window:

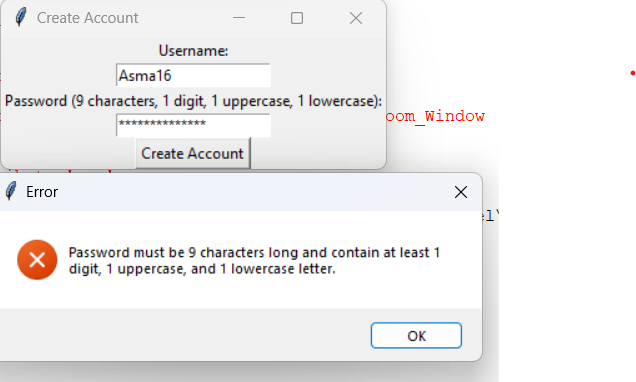


This window will pop when the user clicks the Login button:  


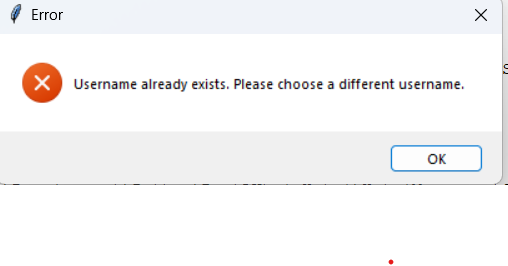
This window will appear when the user clicks on the create account button:



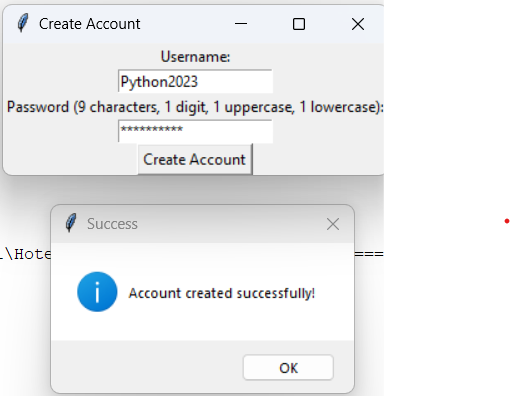
Password Validation message box: (When the password is not in correct format)



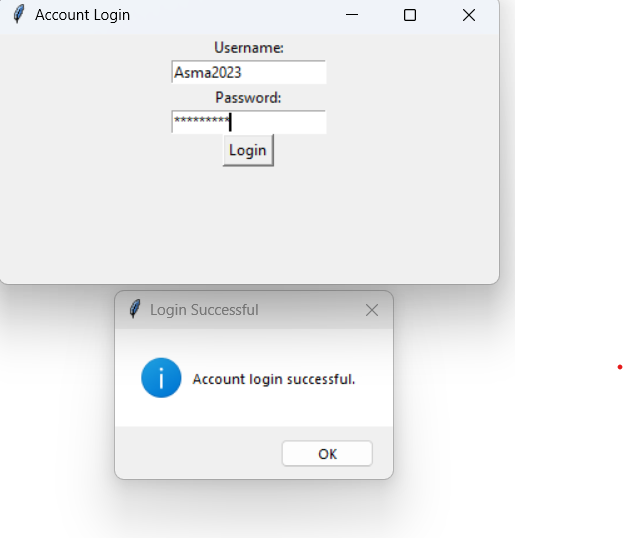
Message Box for duplicate Username:



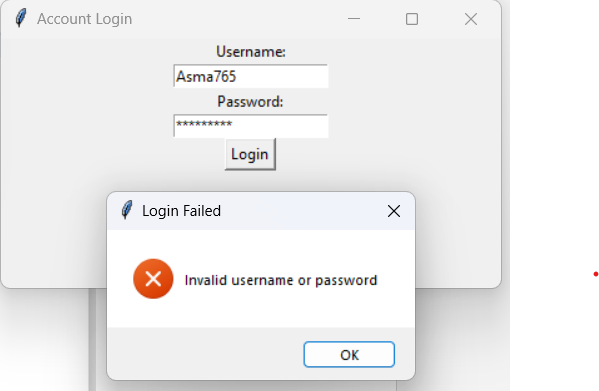
Password Validation message box: (When the password is in correct format)



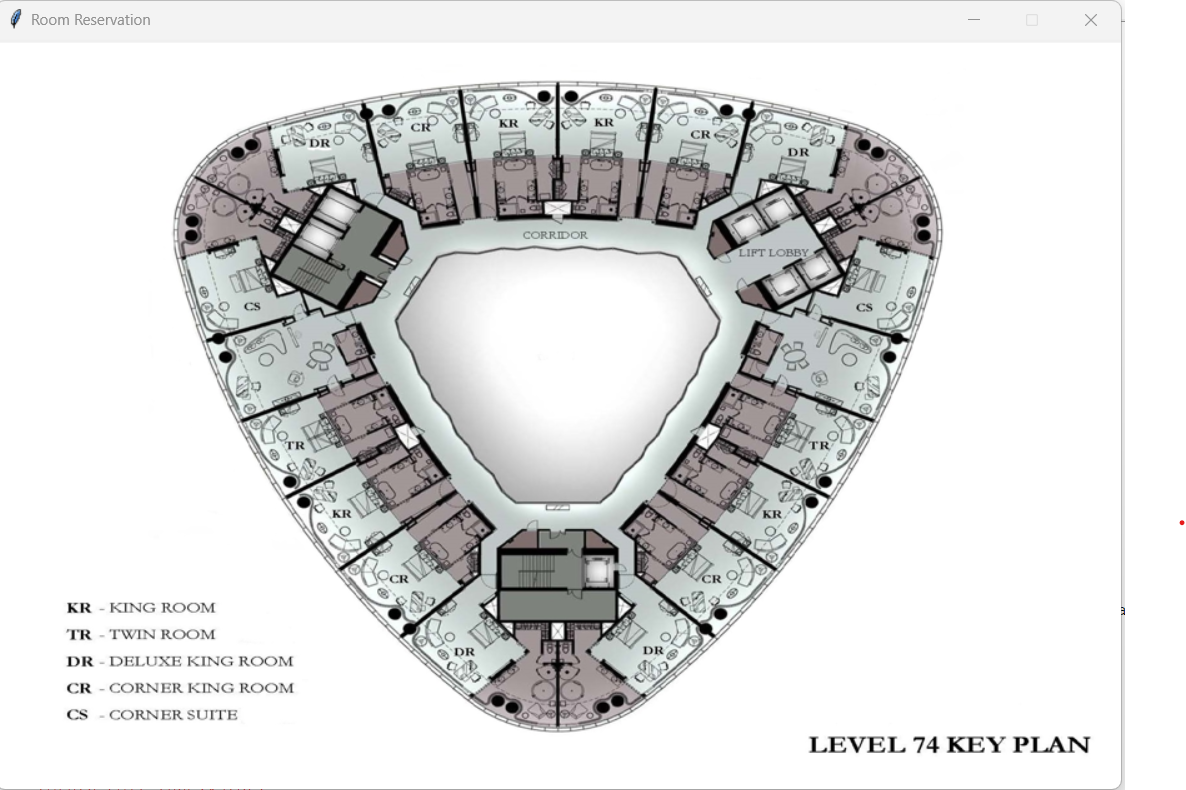
Account Login Window: (When Login is successful)



Account Login Window: (When Login fails)



After Successful Login “Hotel Room Reservation” Window pops up:



* **Design and Development Document for Hotel Management System**

#### Overview

* The Hotel Management System is a Python application with a graphical user interface (GUI) using tkinter. It allows users to create accounts and log in to manage hotel rooms.

#### Code Structure

* **Hotel Class**
* **Initialization**:
* Initializes the main window with a title and fixed size.
* Creates UI elements: title label, "Create Account" button, "Login" button, and a background image.
* **Functionality**
* **create\_account**:
* Handles the "Create Account" button click, disables the button, and opens an account creation window.
* **login**:
* Creates an instance of the **LoginManager** class.
* Sets an initial username and password (you can change these).
* Runs the login manager to display the login GUI.
* **Main Application**
* Creates an instance of the **Hotel** class to start the application.
* Runs the tkinter main event loop to display the GUI.

**Summary:**

This code defines the Hotel Management System, including the main application window, user account creation, and login functionality. The GUI allows users to create accounts and manage hotel rooms. The **Hotel** class initializes the main window and handles user interactions.

**Hotel Management System File Code:**

import tkinter as tk

from PIL import Image, ImageTk

from LoginModule import LoginManager

from CreateAccountModule import CreateAccountWindow

import csv

#Main Window Display

class Hotel:

def \_\_init\_\_(self):

self.main\_win = tk.Tk()

self.main\_win.title("Hotel Room Management")

# Set a fixed window size (e.g., 650x250)

self.main\_win.geometry("650x250")

# Make the window non-resizable

self.main\_win.resizable(False, False)

# Configure columns if needed

self.main\_win.columnconfigure(0, minsize=50)

self.main\_win.columnconfigure(1, minsize=150)

self.main\_win.columnconfigure(2, minsize=250)

self.main\_win.columnconfigure(3, minsize=150)

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#Creating Label of Main Window

self.title\_label = tk.Label(text='Hotel Management System', font=("Helvetica", 15), fg="navy")

self.title\_label.grid(row=0, column=1, columnspan=3) # Center the label in columns 1, 2, and 3

self.create\_account\_button = tk.Button(text="Create Account", width=15, font=("Helvetica", 10), command=self.create\_account)

self.create\_account\_button.grid(row=1, column=1, sticky="nsew") # Position the "Create Account" button in column 1

self.login\_button = tk.Button(text="Login", width=10, font=("Helvetica", 10), command=self.login)

self.login\_button.grid(row=1, column=2, sticky="nsew") # Position the "Login" button in column 2

self.quit\_button = tk.Button(text="Cancel", width=10, font=("Helvetica", 10), command=self.main\_win.destroy)

self.quit\_button.grid(row=1, column=3, sticky="nsew") # Position the "Cancel" button in column 3

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Load and display the image on main window

img = Image.open(r'C:\Users\\Asma\\Desktop\\Asma.gif')

img = img.resize((650, 250))

img = ImageTk.PhotoImage(img)

self.labelGIF = tk.Label(image=img)

self.labelGIF.image = img

self.labelGIF.place(x=50, y=60)

tk.mainloop()

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#Create Account Button Module

def create\_account(self):

# Replace this with the code to handle the "Create Account" button click

print("Create Account button clicked")

# Disable the "Create Account" button once clicked

self.create\_account\_button.configure(state="disabled")

self.create\_account\_window = CreateAccountWindow()

self.create\_account\_window.run()

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#Login Button functionality

def login(self):

# Create an instance of the LoginManager class

self.login\_manager\_instance = LoginManager()

#self.login\_manager\_instance.configure(state="disabled")

# Set initial username and password (you can change these)

self.login\_manager\_instance.AddLogin("admin", "password")

# Run the login manager and display the login

print("calling run")

self.login\_manager\_instance.run()

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Validate Login Information \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def validate\_login(self):

# Get the entered username and password

username = self.username\_entry.get()

password = self.password\_entry.get()

# Load user data from the txt file

with open('usernames.txt', mode='r') as file:

csv\_reader = csv.reader(file)

for row in csv\_reader:

saved\_username, saved\_password = row

if username == saved\_username and password == saved\_password:

print(saved\_username + "," + saved\_password)

print("Login successful")

return True

print("Invalid username or password")

return False

print("About to create hotel object");

Hotel\_Mnagement\_System = Hotel()

* **Design and Development Document for Create Account Window**

#### Overview

* The provided Python code defines a graphical user interface (GUI) using tkinter for creating user accounts. It includes fields for entering a username and password, along with validation checks for the password's complexity. Usernames and passwords are saved in separate text files.

#### Code Structure

* **CreateAccountWindow Class**
* **Initialization**:
* Initializes a new window for creating accounts (Toplevel).
* Provides labels and entry fields for entering a username and password.
* Includes buttons for creating a username and password.
* **Functionality**
* **is\_valid\_password**:
* Validates the password for complexity using regular expressions.
* **is\_username\_unique**:
* Checks if the entered username is unique by comparing it to existing usernames in a text file.
* **create\_account**:
* Collects the entered username and password.
* Validates the password and username.
* Saves the username and password to separate text files.
* Displays success or error messages accordingly.
* **run**:
* Starts the Tkinter main loop to display the GUI.

**Summary:**

This code defines a user account creation window with password complexity validation and username uniqueness checks. Usernames and passwords are saved to separate text files. The **CreateAccountWindow** class encapsulates this functionality and facilitates interaction with the GUI.

# **Create Account Window Code:**

i import tkinter as tk

from tkinter import messagebox

import re # Import the 're' module for regular expressions

class CreateAccountWindow:

def \_\_init\_\_(self):

print("CreateAccountWindow Constructor called.")

self.create\_account\_window = tk.Toplevel()

self.create\_account\_window.title("Create Account")

# Labels and Entry fields for username and password

tk.Label(self.create\_account\_window, text="Username:").pack()

self.username\_entry = tk.Entry(self.create\_account\_window)

self.username\_entry.pack()

tk.Label(self.create\_account\_window, text="Password (9 characters, 1 digit, 1 uppercase, 1 lowercase):").pack()

self.password\_entry = tk.Entry(self.create\_account\_window, show="\*")

self.password\_entry.pack()

# Button to create the account

tk.Button(self.create\_account\_window, text="Create Account", command=self.create\_account).pack()

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*File for Username and Password \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def is\_valid\_password(self, password):

return re.match(r'^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)[a-zA-Z\d]{9,}$', password)

def is\_username\_unique(self, username):

with open("usernames.txt", "r") as file:

lines = file.readlines()

for line in lines:

print(" line read from the file is: " + line)

if len(line) >0:

print("Length of the line is " + str(len(line)))

existing\_username, \_ = line.strip().split(',')

if existing\_username == username:

return False

return True

def create\_account(self):

username = self.username\_entry.get()

password = self.password\_entry.get()

if not self.is\_valid\_password(password):

messagebox.showerror("Error", "Password must be 9 characters long and contain at least 1 digit, 1 uppercase, and 1 lowercase letter.")

elif not self.is\_username\_unique(username):

messagebox.showerror("Error", "Username already exists. Please choose a different username.")

else:

try:

# Save the username and password as a pair to the accounts.txt file

with open("usernames.txt", "a") as file:

file.write(f"{username},{password}\n")

messagebox.showinfo("Success", "Account created successfully!")

self.create\_account\_window.destroy()

except Exception as e:

messagebox.showerror("Error", "Failed to save account: " + str(e))

def run(self):

# Start the Tkinter main loop

print("CreateAccountModule Run method called");

self.create\_account\_window.mainloop()

### **Design and Development Document for Login Module:**

#### Overview

The provided Python code presents a Login Manager class that creates a graphical user interface (GUI) window for user authentication using the tkinter library. Users input their username and password, and the system checks if the provided credentials match the stored values. The class facilitates adding and verifying login credentials.

#### Code Structure

**LoginManager Class**

* **Initialization**:
* Initializes a Tkinter window for account login.
* Provides labels and entry widgets for entering a username and password.
* Includes a login button for initiating the login process.

**Functionality**

* **verify\_login**:
* Collects entered username and password and verifies them using the **VerifyLogin** method.
* **VerifyLogin**:
* Compares the entered credentials with the stored username and password and returns **True** if they match.
* **AddLogin**:
* Stores the username and password for later verification.
* **run**:
* Starts the Tkinter main loop to display the GUI.

**Summary:**

The code provided defines a simple login management system with a graphical user interface. It allows users to input their credentials, verify them, and provides an option to add new login credentials for later use. The **LoginManager** class encapsulates this functionality and facilitates interaction with the GUI.

**Login Module Code:**

import tkinter as tk

import csv

from tkinter import messagebox

from HotelRoomReservationModule import ReserveRoomWindow

class LoginManager:

def \_\_init\_\_(self):

print("Login Manager Constructor");

self.usernameStored = ""

self.passwordStored = ""

self.AccountLogin = tk.Tk() # Create a Tkinter AccountLogin window

self.AccountLogin.title("Account Login")

# Set window size

self.AccountLogin.geometry("400x200")

# Create labels and entry widgets for username and password

tk.Label(self.AccountLogin, text="Username:").pack()

self.username\_entry = tk.Entry(self.AccountLogin)

self.username\_entry.pack()

tk.Label(self.AccountLogin, text="Password:").pack()

self.password\_entry = tk.Entry(self.AccountLogin, show="\*") # Use show="\*" to hide the password

self.password\_entry.pack()

# Create a login button

tk.Button(self.AccountLogin, text="Login", command=self.verify\_login).pack()

def verify\_login(self):

# Get the entered username and password

username = self.username\_entry.get()

password = self.password\_entry.get()

with open('usernames.txt', mode='r') as file:

csv\_reader = csv.reader(file)

for row in csv\_reader:

if len(row) >= 2:

saved\_username, saved\_password = row

if username == saved\_username and password == saved\_password:

messagebox.showinfo("Login Successful", "Account login successful.")

# Close the login window

print("About to destroy the login window")

self.AccountLogin.destroy()

# Open a new window upon successful login

self.Display\_Reserve\_Room\_Window()

return

messagebox.showerror("Login Failed", "Invalid username or password")

def AddLogin(self, username, password):

self.usernameStored = username

self.passwordStored = password

def Display\_Reserve\_Room\_Window(self):

self.ReservationWindow=ReserveRoomWindow()

self.ReservationWindow.run()

def run(self):

# Start the Tkinter main loop

self.AccountLogin.mainloop()

### **Design and Development Document for ReserveRoomWindow**

#### **Overview**

The provided Python code defines a **ReserveRoomWindow** class that creates a graphical user interface (GUI) window for reserving hotel rooms. This class is used to initialize and display a reservation window, which includes an image of the hotel.

#### Code Structure

**ReserveRoomWindow Class**

* **Initialization**:
* Initializes a child window (Toplevel) for room reservation.
* Sets the window title, size, and makes it non-resizable.
* Loads and displays an image of the hotel as the window background.

#### **Room Reservation Window Code:**

import tkinter as tk

from PIL import Image, ImageTk

from CreateAccountModule import CreateAccountWindow

import csv

class ReserveRoomWindow:

def \_\_init\_\_(self):

print("ReserveRoomWindow Constructor called")

# get the hotel image

self.ReservationWindow = tk.Toplevel();#Here we use TopLevel() to create the window. Use TK() to create the main window,

#and use tk.TopLevel() to use child windows.

#Also do not call main loop on child windows.

#if these rules are not followed, program will not work.

#TK() is used to create the main program window. Only call mainloop method

#on main program window.

self.ReservationWindow.title("Room Reservation")

self.ReservationWindow.geometry("900x600")

self.ReservationWindow.resizable(False, False)

# Load the image

try:

img = Image.open(r'C:\Users\Asma\Desktop\Hotel.jpg')

img = img.resize((900, 600))

img = ImageTk.PhotoImage(img)

self.labelGIF = tk.Label(self.ReservationWindow, image=img)

self.labelGIF.image = img

self.labelGIF.place(x= 1, y=1)

except Exception as e:

print("Error loading image:", str(e))