FACULTY OF COMPUTERING, INFORMATIC & MATHEMATICS

UNIVERSITI TEKNOLOGI MARA

MERBOK, KEDAH

Diploma in Library Informatics

(CDIM144)

Programming for Libraries

(IML208)

INDIVIDUAL PROJECT:

Spa Appointment System

By:

Dinah Fathiney binti Azman (2023803232)

Class: KCDIM1443B

Prepared for:

Sir Mohd Firdaus bin Mohd Helmi

Submission Date:

17th December 2024

INDIVIDUAL ASSIGNMENT: SPA APPOINTMENT SYSTEM

DINAH FATHINEY BINTI AZMAN

2023803232

KCDIM1443B

UNIVERSITI TEKNOLOGI MARA

(UiTM), KEDAH, KAMPUS SUNGAI PETANI

SCHOOL OF INFORMATION SCIENCE

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

DIPLOMA IN LIBRARY INFORMATICS

2024

TABLE OF CONTENT

ACKNOWLEDGEMENT

Assalamualaikum w.b.t,

Alhamdulillah, I would like to thank God for giving me the strength to complete the assignment smoothly. And because of His grace, I was encouraged to continue the assignment until I succeeded.

Me, as a part of the students from CDIM144 Library Informatics, want to convey my sincere gratitude to Sir Mohd Firdaus bin Mohd Helmi, my lecturer for his entire attention and infinite help in completing this assignment. Without his guidance, I could not even finish this assignment completely all by myself.

Apart from that, I would like to thank all my beloved lecturer and classmates for helping me without any hesitation throughout this assignment. Their help in answering the questions truly gives clarity to all my doubts and issues.

Last but not least, I would like to thank my family for their continuous support until I accomplish this assignment. With their support and motivation, I was determined to not give up halfway and finish this assignment easily. Also, all that involved directly or indirectly in this assignment.

أونيۏرسيتي تيكنولوڬي مارا
UNIVERSITI TEKNOLOGI MARA

STUDENT PLEDGE OF ACADEMIC INTEGRITY

As a student of Universiti Teknologi MARA (UiTM), it is my responsibility to act in accordance with UiTM's academic assessment and evaluation policy. I hereby pledge to act and uphold academic integrity and pursue scholarly activities in UiTM with honesty and responsible manner. I will not engage or tolerate acts of academic dishonesty, academic misconduct, or academic fraud including but not limited to:

a. Cheating: Using or attempt to use any unauthorized device, assistance, sources, practice or materials while completing academic assessments. This include but not limited to copying from another, allowing others to copy, unauthorized collaboration on an assignment or open book tests, or engaging in any act or conduct that can be construed as cheating.

b. Plagiarism: Using or attempts to use the work of others (ideas, design, words, art, music, etc.) without acknowledging the source; using or purchasing materials prepared by another person or agency or engaging in other behavior that a reasonable person would consider as plagiarism.

c. Fabrication: Falsifying data, information, or citations in any academic assessment and evaluation.

d. Deception: Providing false information with intend to deceive an instructor concerning any academic assessment and evaluation.

e. Furnishing false information: Providing false information or false representation to any UiTM official, instructor, or office.

With this pledge, I am fully aware that I am obliged to conduct myself with utmost honesty and integrity. I fully understand that a disciplinary action can be taken against me if I, in any manner, violate this pledge.

[illegible]

Name: Dinah Fathiney binti Azman

Matric Number: 2023803232

Course Code: CDIM144

Programme code: IML 153

Faculty / Campus: Faculty of Computering, Mathematics and Informatics

1.0 Prompt Data:

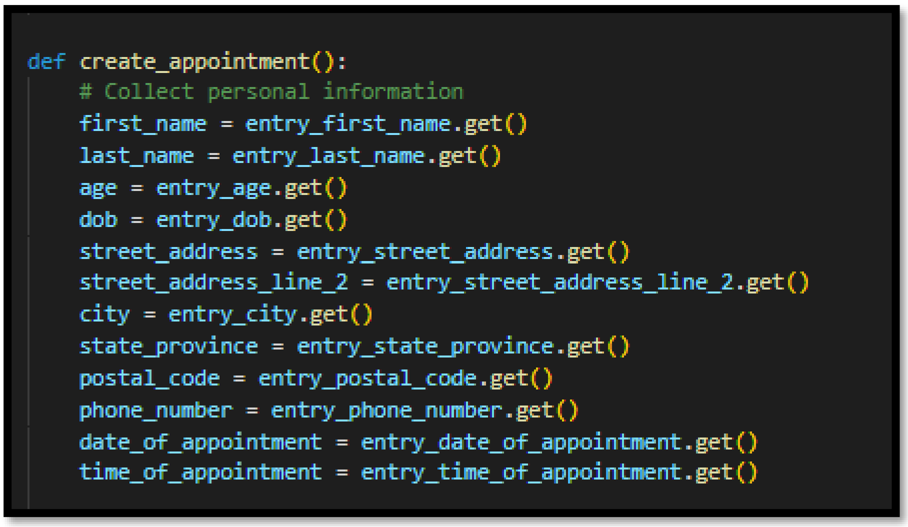
1. First name.
2. Last name.
3. Age.
4. Date of birth in day, month, year.
5. Street Adress.
6. Street Adress Line 2.
7. City.
8. State/Province.
9. Postal Code.
10. Phone Number.
11. Date of appointment in day, month, year.
12. Time of appointment in hour, minute.
13. The spa packages that are available, which is packages 1, 2, 3 or 4. Option 4 is custom packages.

2.0 Function:

i. CREATE: Client can create an appointment.

   Clients can create an appointment by entering their information based on the prompt data that has been stated before:

   - First name.
   - Last name.
   - Age.
   - Date of birth in day, month, year.
   - Street Address.
   - Street Address Line 2.
   - City.
   - State/Province.
   - Postal Code.
   - Phone Number.
   - Date of appointment in day, month, year.
   - Time of appointment in day, month, year.
   - Customers can create information by selecting the spa packages that are available, which is packages 1, 2, 3 or 4. Option 4 is custom packages, and customers can create information by customs of their own packages.



```python
def create_appointment():
    # Collect personal information
    first_name = entry_first_name.get()
    last_name = entry_last_name.get()
    age = entry_age.get()
    dob = entry_dob.get()
    street_address = entry_street_address.get()
    street_address_line_2 = entry_street_address_line_2.get()
    city = entry_city.get()
    state_province = entry_state_province.get()
    postal_code = entry_postal_code.get()
    phone_number = entry_phone_number.get()
    date_of_appointment = entry_date_of_appointment.get()
    time_of_appointment = entry_time_of_appointment.get()
```

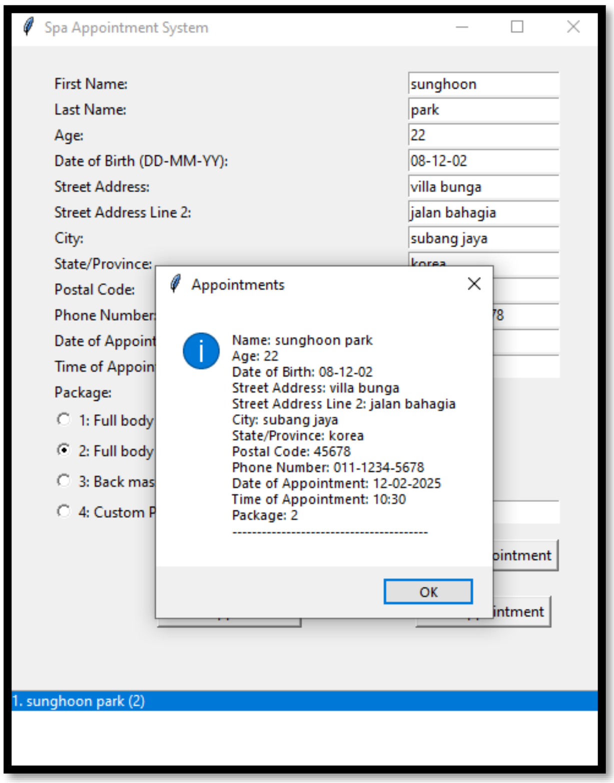
Figure 1.0 Create function

Based on figure 1.0, customers can create information when booking an appointment by entering their personal information. After completing all of the personal information, customers will be able to create the spa booking appointment by simply clicking on the creation appointment button that has been provided at the bottom of the booking system.

ii. READ: Clients and receptionists of the spa can read the appointment. The information that can be read are:

- First name.
- Last name.
- Age.
- Date of birth in day, month, year.
- Street Address.
- Street Address Line 2.
- City.
- State/Province.
- Postal Code.
- Phone Number.
- Date of appointment in day, month, year.
- Time of appointment in day, month, year.
- The package that customers have chosen.

This is the result of the spa appointment booking system that can be read:


Spa Appointment System

First Name:	sunghoon
Last Name:	park
Age:	22
Date of Birth (DD-MM-YY):	08-12-02
Street Address:	villa bunga
Street Address Line 2:	jalan bahagia
City:	subang jaya
State/Province:	korea
Postal Code:	
Phone Number	78
Date of Appoint	
Time of Appoin	
Package:	

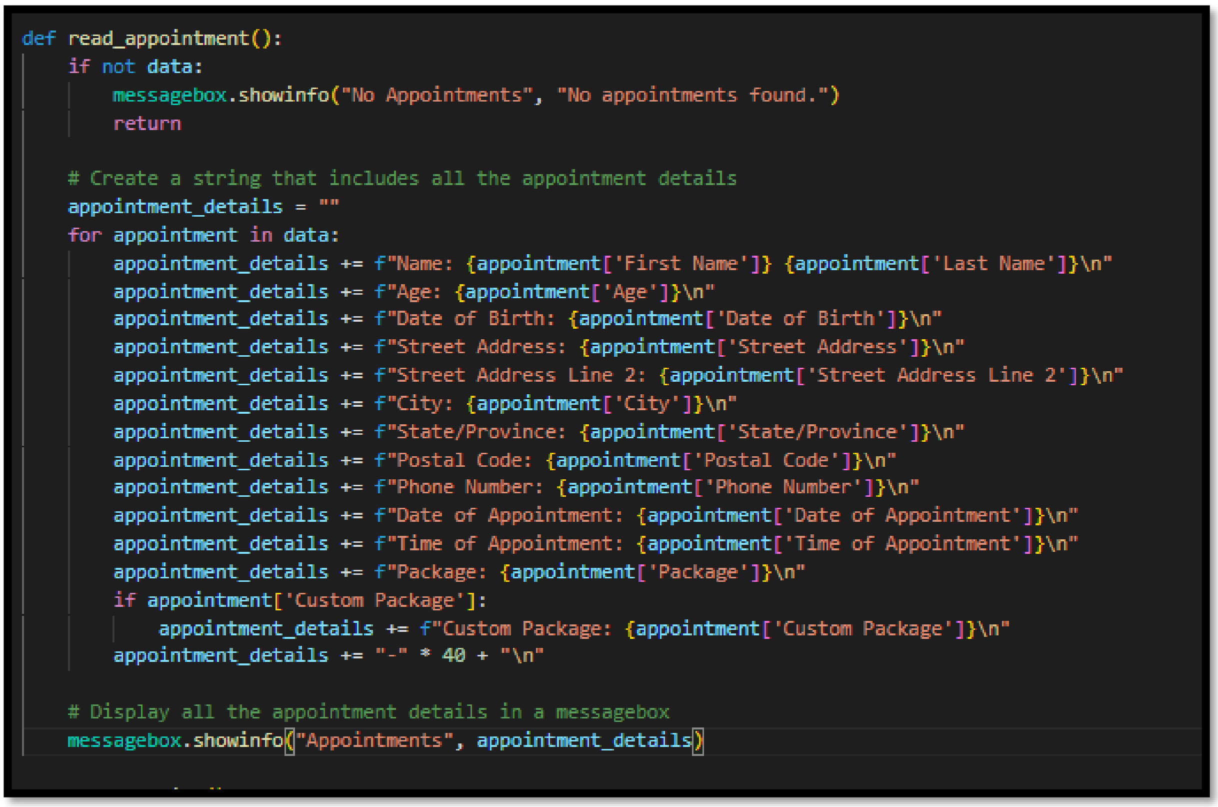
( ) 1: Full body
[✓ 2: Full body]
( ) 3: Back mas
( ) 4: Custom P

Appointments

```
Name: sunghoon park
Age: 22
Date of Birth: 08-12-02
Street Address: villa bunga
Street Address Line 2: jalan bahagia
City: subang jaya
State/Province: korea
Postal Code: 45678
Phone Number: 011-1234-5678
Date of Appointment: 12-02-2025
Time of Appointment: 10:30
Package: 2
----------------------------------------
```

1. sunghoon park (2)

Figure 2.0 Information that can be read

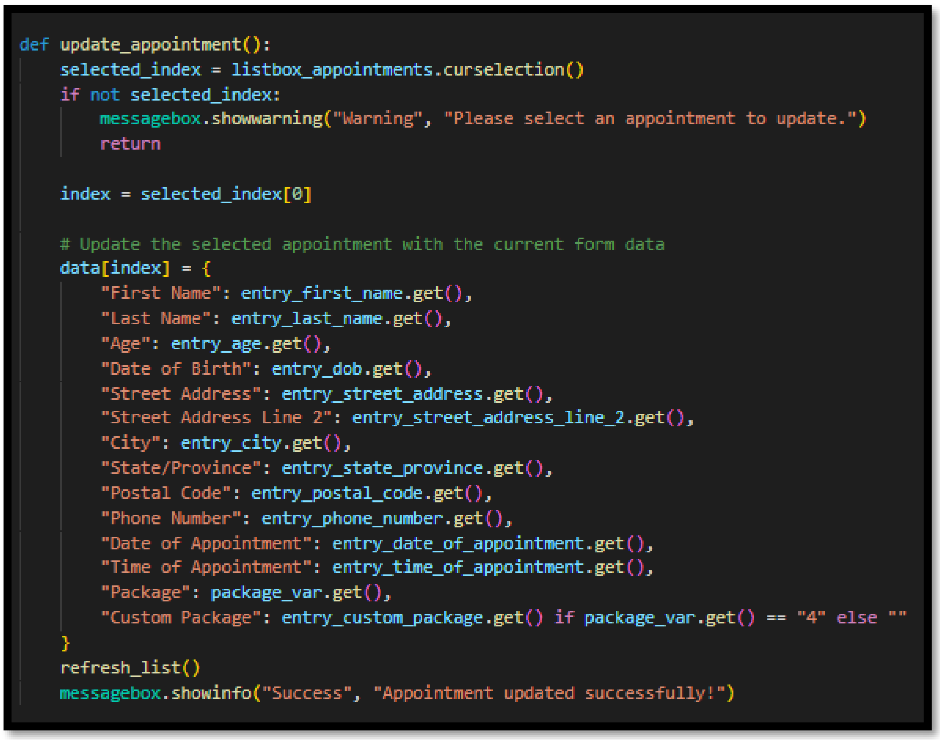


```python
def read_appointment():
    if not data:
        messagebox.showinfo("No Appointments", "No appointments found.")
        return

    # Create a string that includes all the appointment details
    appointment_details = ""
    for appointment in data:
        appointment_details += f"Name: {appointment['First Name']} {appointment['Last Name']}\n"
        appointment_details += f"Age: {appointment['Age']}\n"
        appointment_details += f"Date of Birth: {appointment['Date of Birth']}\n"
        appointment_details += f"Street Address: {appointment['Street Address']}\n"
        appointment_details += f"Street Address Line 2: {appointment['Street Address Line 2']}\n"
        appointment_details += f"City: {appointment['City']}\n"
        appointment_details += f"State/Province: {appointment['State/Province']}\n"
        appointment_details += f"Postal Code: {appointment['Postal Code']}\n"
        appointment_details += f"Phone Number: {appointment['Phone Number']}\n"
        appointment_details += f"Date of Appointment: {appointment['Date of Appointment']}\n"
        appointment_details += f"Time of Appointment: {appointment['Time of Appointment']}\n"
        appointment_details += f"Package: {appointment['Package']}\n"
        if appointment['Custom Package']:
            appointment_details += f"Custom Package: {appointment['Custom Package']}\n"
        appointment_details += "-" * 40 + "\n"

    # Display all the appointment details in a messagebox
    messagebox.showinfo("Appointments", appointment_details)
```

Figure 3.0 Read function

Based on figure 3.0, the code allow customers or spa staff to read the information that has been filled before just by clicking on the appointment stored and then clicking on the read appointment button.

iii. UPDATE: The client can update or change the date of appointment. The information that can be updated is the same as the information that can be read.

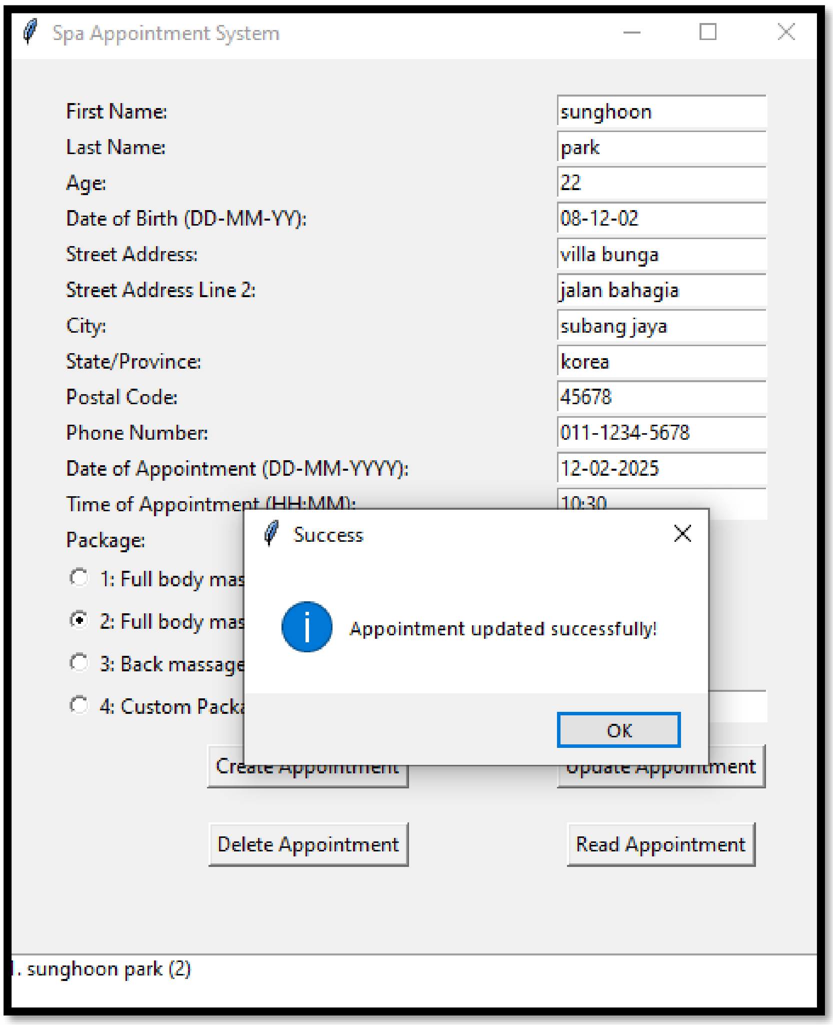


```python
def update_appointment():
    selected_index = listbox_appointments.curselection()
    if not selected_index:
        messagebox.showwarning("Warning", "Please select an appointment to update.")
        return

    index = selected_index[0]

    # Update the selected appointment with the current form data
    data[index] = {
        "First Name": entry_first_name.get(),
        "Last Name": entry_last_name.get(),
        "Age": entry_age.get(),
        "Date of Birth": entry_dob.get(),
        "Street Address": entry_street_address.get(),
        "Street Address Line 2": entry_street_address_line_2.get(),
        "City": entry_city.get(),
        "State/Province": entry_state_province.get(),
        "Postal Code": entry_postal_code.get(),
        "Phone Number": entry_phone_number.get(),
        "Date of Appointment": entry_date_of_appointment.get(),
        "Time of Appointment": entry_time_of_appointment.get(),
        "Package": package_var.get(),
        "Custom Package": entry_custom_package.get() if package_var.get() == "4" else ""
    }
    refresh_list()
    messagebox.showinfo("Success", "Appointment updated successfully!")
```

Figure 4.0 Update function

Based on figure 4.0, customers will be able to update their personal information if they want to change any information by simply clicking back at their stored information and clicking the update button.

This is the result of the information has been changed or updated:



Spa Appointment System

First Name:	sunghoon
Last Name:	park
Age:	22
Date of Birth (DD-MM-YY):	08-12-02
Street Address:	villa bunga
Street Address Line 2:	jalan bahagia
City:	subang jaya
State/Province:	korea
Postal Code:	45678
Phone Number:	011-1234-5678
Date of Appointment (DD-MM-YYYY):	12-02-2025
Time of Appointment (HH:MM):	10:30

Package:

1: Full body mas
[✓ 2: Full body mas]
3: Back massage
4: Custom Pack

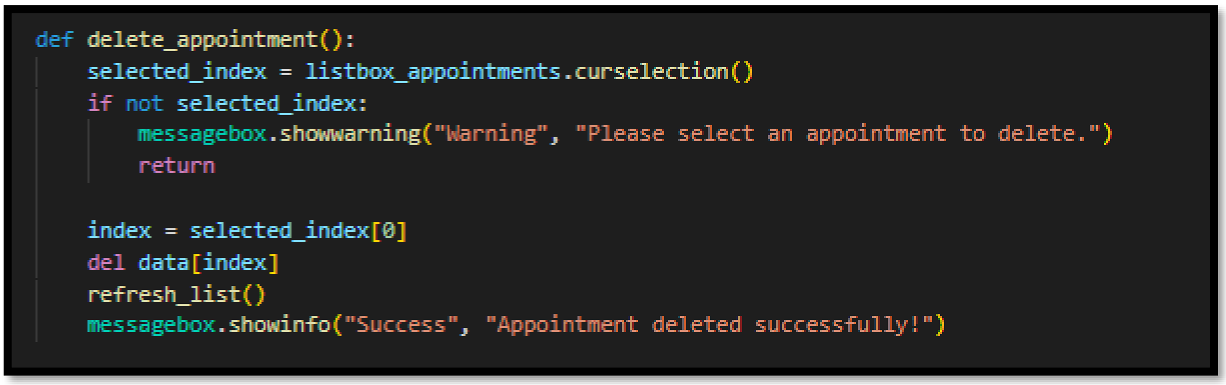
Success

Appointment updated successfully!

1. sunghoon park (2)

Figure 5.0 Result for update function

iv. DELETE: The client can delete or cancel the appointment. Information that can be deleted is the same as the information that can be read and updated.

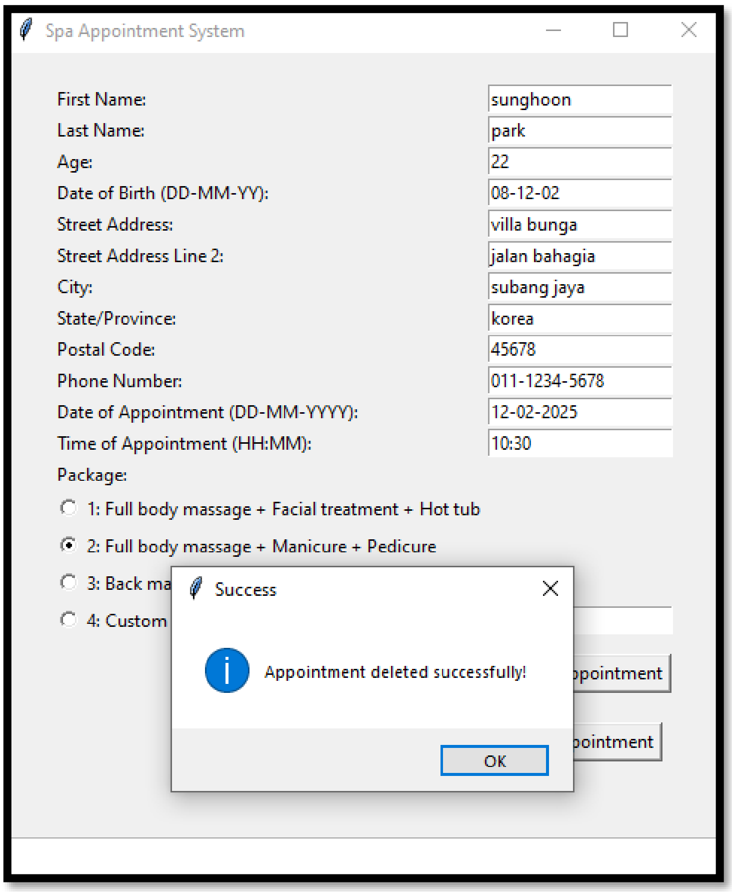


```python
def delete_appointment():
    selected_index = listbox_appointments.curselection()
    if not selected_index:
        messagebox.showwarning("Warning", "Please select an appointment to delete.")
        return

    index = selected_index[0]
    del data[index]
    refresh_list()
    messagebox.showinfo("Success", "Appointment deleted successfully!")
```

Figure 6.0 Delete function

Based on Figure 6.0, the code allow customers to delete the appointment that has been stored before if they want to cancel their appointment.

This is the result after the appointment has been deleted:



Spa Appointment System

First Name:	sunghoon
Last Name:	park
Age:	22
Date of Birth (DD-MM-YY):	08-12-02
Street Address:	villa bunga
Street Address Line 2:	jalan bahagia
City:	subang jaya
State/Province:	korea
Postal Code:	45678
Phone Number:	011-1234-5678
Date of Appointment (DD-MM-YYYY):	12-02-2025
Time of Appointment (HH:MM):	10:30

Package:

1: Full body massage + Facial treatment + Hot tub

[✓ 2: Full body massage + Manicure + Pedicure]

3: Back ma

4: Custom

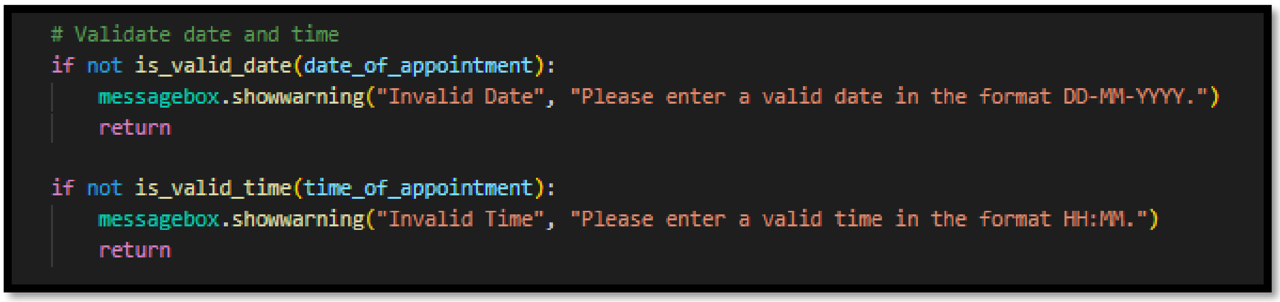
Success

Appointment deleted successfully!

Figure 7.0 Result after the appointment has been deleted

3.0 Conditional statement: Yes

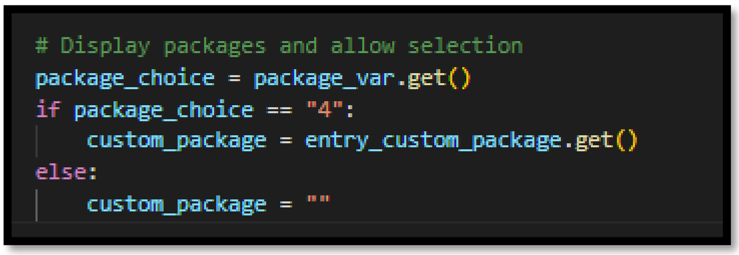
Yes, my system have if and else statement.



```
# Validate date and time
if not is_valid_date(date_of_appointment):
    messagebox.showwarning("Invalid Date", "Please enter a valid date in the format DD-MM-YYYY.")
    return

if not is_valid_time(time_of_appointment):
    messagebox.showwarning("Invalid Time", "Please enter a valid time in the format HH:MM.")
    return
```

Figure 8.0 If statement

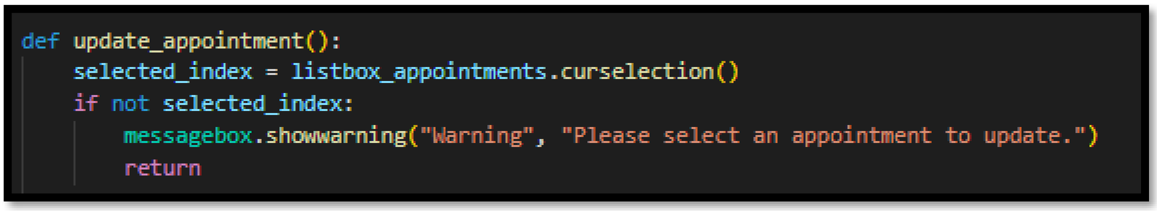
Based on Figure 8.0, the first If statement verifies the validity of the date that was entered in the date_of_appointment field. It seems likely that the is_valid_date() function uses logic (like a regular expression) to determine whether the date is formatted correctly (DD-MM-YYYY). The code inside the if block will run if the date is invalid because the not operator inverts the result of is_valid_date().
The second If statement verifies the validity of the time supplied in the time_of_appointment field, just like the date validation does. The function is_valid_time() determines whether the time is formatted correctly (HH:MM). The code inside the if block will run if the time is invalid.



```
# Display packages and allow selection
package_choice = package_var.get()
if package_choice == "4":
    custom_package = entry_custom_package.get()
else:
    custom_package = ""
```

Figure 9.0 If and else statement

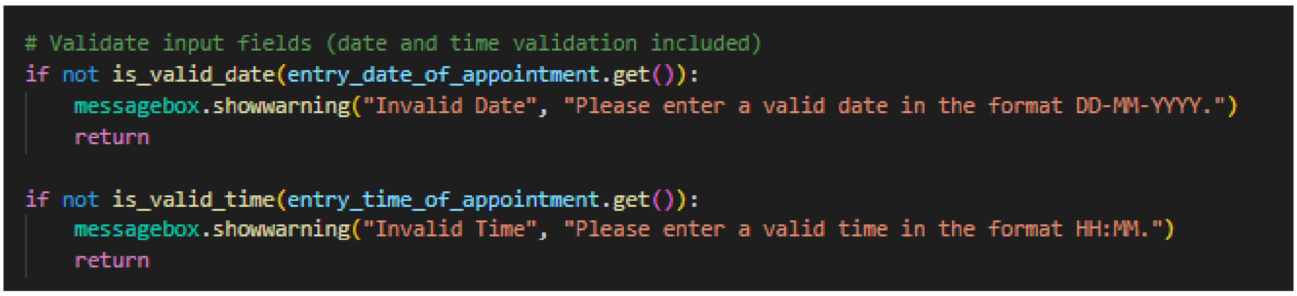
Based on the figure 9.0, the If statement show the function if a customer choose package 4 which is custom package means customer want to custom their packages besides the 3 other packages. For else statement, the code sets custom_package to an empty string (" ") since the custom package is meaningless if the user did not choose option 4.



```
def update_appointment():
    selected_index = listbox_appointments.curselection()
    if not selected_index:
        messagebox.showwarning("Warning", "Please select an appointment to update.")
        return
```

Figure 10.0 If statement

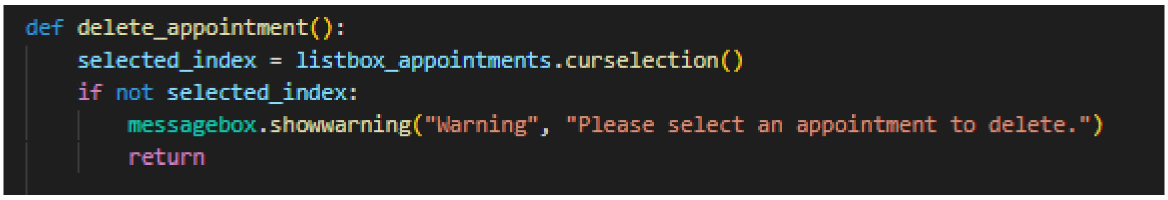
Based on figure 10.0, the If statement show the function employed to verify whether the user has chosen an appointment from the listbox_appointments before updating it.



```
# Validate input fields (date and time validation included)
if not is_valid_date(entry_date_of_appointment.get()):
    messagebox.showwarning("Invalid Date", "Please enter a valid date in the format DD-MM-YYYY.")
    return

if not is_valid_time(entry_time_of_appointment.get()):
    messagebox.showwarning("Invalid Time", "Please enter a valid time in the format HH:MM.")
    return
```

Figure 11.0 If statement

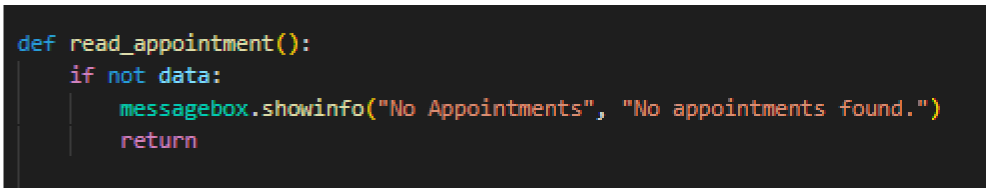
Based on figure 11.0, This code's if statements serve as conditional checks to verify the user-inputted date and time. Before continuing, they make that the input complies with the necessary format.



```python
def delete_appointment():
    selected_index = listbox_appointments.curselection()
    if not selected_index:
        messagebox.showwarning("Warning", "Please select an appointment to delete.")
        return
```

Figure 12.0 If statement

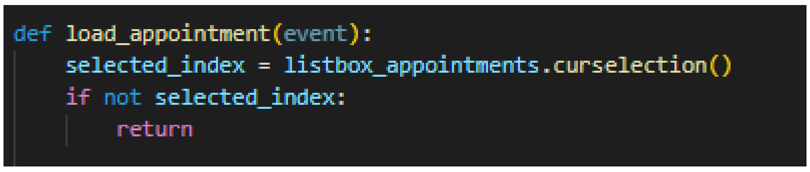
Based on figure 12.0, the update_appointment function's if statement serves a similar role as the delete_appointment function's. Before deleting an appointment, it makes sure that it has been chosen.



```python
def read_appointment():
    if not data:
        messagebox.showinfo("No Appointments", "No appointments found.")
        return
```

Figure 13.0 If statement

Based on figure 13.0, The read_appointment() function uses an if statement to determine whether the data list, which contains all appointment details, is empty.



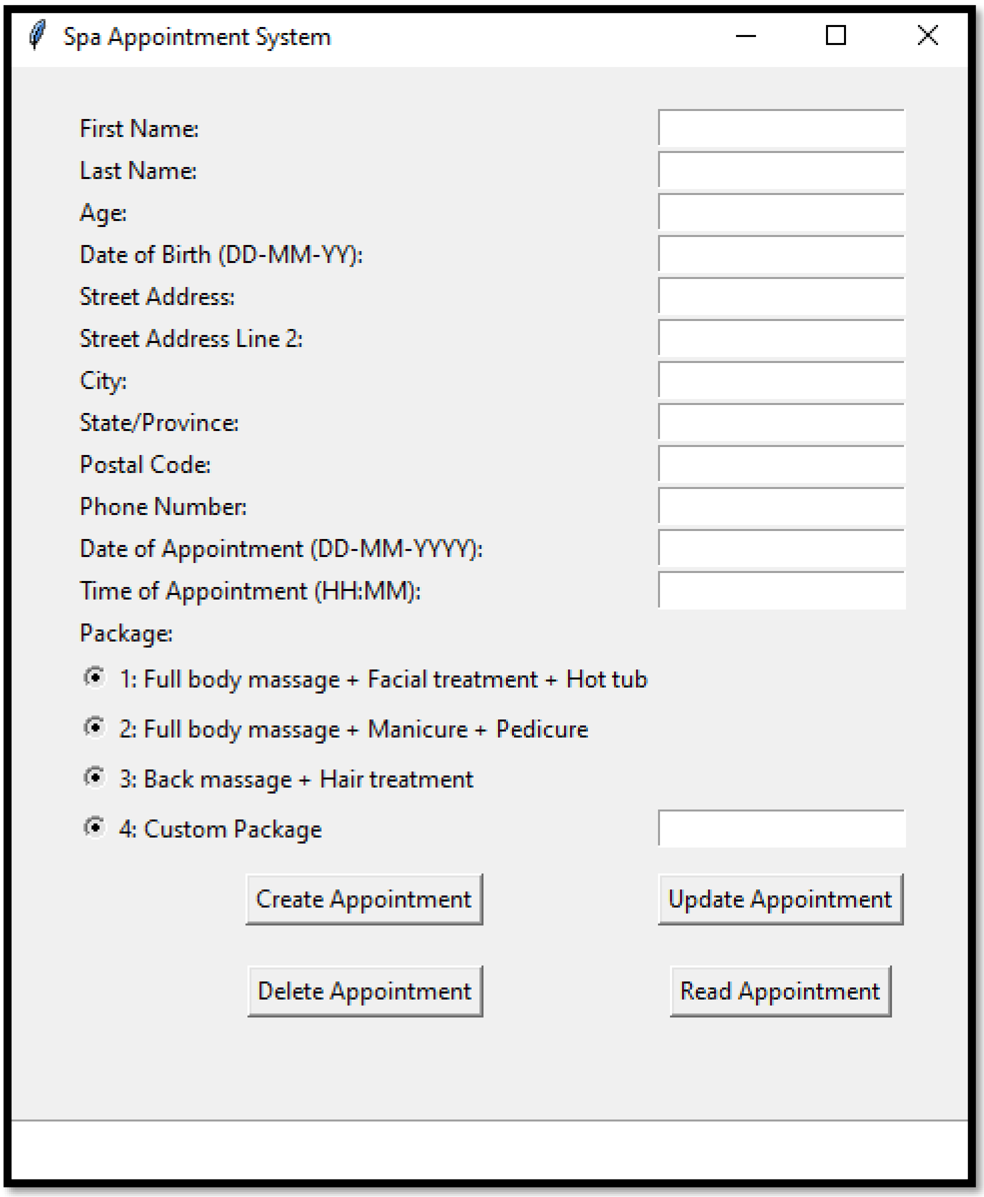
```python
def load_appointment(event):
    selected_index = listbox_appointments.curselection()
    if not selected_index:
        return
```

Figure 14.0 If statement

Based on figure 14.0, the if statement means before trying to load an appointment's details into the form, the load_appointment function's if statement makes sure that the appointment is selected.

4.0 GUI

Yes, I have the GUI for my system:



Spa Appointment System

First Name:

Last Name:

Age:

Date of Birth (DD-MM-YY):

Street Address:

Street Address Line 2:

City:

State/Province:

Postal Code:

Phone Number:

Date of Appointment (DD-MM-YYYY):

Time of Appointment (HH:MM):

Package:

- 1: Full body massage + Facial treatment + Hot tub
- 2: Full body massage + Manicure + Pedicure
- 3: Back massage + Hair treatment
- 4: Custom Package

Figure 15.0 GUI

5.0 Strength

Improved client experience and saves time

By offering a simple online booking system, my spa appointment system intends to improve the customer experience. This system minimizes the need for customers to physically visit our outlet in order to make an appointment. Instead, they can do it right away through their mobile devices. This saves clients a significant amount of time and makes the whole process more convenient. From the convenience of their homes, they may quickly and simply schedule appointments without having to leave the house or go anywhere.

Increase staff productivity and efficient scheduling

By making administrative activities like scheduling, record keeping, and inventory management easier, online appointment booking systems significantly increase employee productivity. Based on client references, service duration, and staff availability, the system can provide an appointment date that available and not available for appointment. Additionally, it will reduce schedule issues and avoid double booking. As a result, our employees will be less stressed and more productive.

6.0 Kaizen: room for improvement of my system

Initital setup cost

The system will require an initial setup cost during its development, which could create significant challenges for us and might lead to an increase in our overall budget. Along with these expenses, we will also need to set aside money for staff training so they can manage and upkeep the system efficiently.

Technical issue on the system

Any online system, including our booking system, is at risk of technical problems. Potential system crashes, software glitches, and trouble connecting to the internet are some of the examples of these problems. These issues can cause problems with the booking process's efficient operation, making it more difficult for customers to make appointments and possibly worsening the user experience as a whole.