Laboratory Activity 5 - Introduction to Event Handling in GUI Development	
BONIFACIO, NYKO ADREIN L.	10/21/2024
Course/Section: CPE009B - CPE21S4	Prof. Sayo

6. Supplementary Activity

Using the simple Account Registration system created in the previous laboratory activity. Create the functionality that will register the account by saving it to a Database using sqlitedict or a .csv or .txt file. The GUI program should not allow the registration to proceed if there is a field that has an empty value and notify of the missing values using a message box. Once the registration is successful a message should appear that would inform the user that the registration was successful. Use the appropriate symbol in making the message box.

```
from tkinter import *
from tkinter import messagebox
import csv
import os

class RegistrationWindow:
   def __init__(self, win):
      self.label1 = Label(win, text="Account Registration", font=("Impact", 25), bg="Orange",
fg="White")
      self.label1.place(x=80, y=10)

      fields = ["First Name:", "Last Name:", "Username:", "Password:", "Email Address:", "Contact
Number:"]
      self.entries = []
      y_position = 90

      for i, field in enumerate(fields):
         Label(win, text=field, font=("Arial", 14), anchor='w').place(x=50, y=y_position)
         entry = Entry(win, bd=5)
         entry.place(x=200, y=y_position)
         self.entries.append(entry)
         y_position += 40

      self.submit_btn = Button(win, text="Submit", bd=4, bg="LightGreen", font=("Arial", 14),
command=self.submit)
      self.submit_btn.place(x=100, y=y_position + 20)

      self.clear_btn = Button(win, text="Clear", bd=4, bg="LightCoral", font=("Arial", 14),
command=self.clear)
      self.clear_btn.place(x=200, y=y_position + 20)
```

```
    def submit(self):
        for i, entry in enumerate(self.entries):
            if not entry.get().strip():
                messagebox.showerror("Input Error", f"Make sure nothing is skipped.")
                return

        data = {field: entry.get() for field, entry in zip(["First Name", "Last Name", "Username",
"Password", "Email Address", "Contact Number"], self.entries)}


        file_exists = os.path.isfile('registrations.csv')
        with open('registrations.csv', mode='a', newline='') as file:
            writer = csv.writer(file)
            if not file_exists:
                writer.writerow(["First Name", "Last Name", "Username", "Password", "Email Address",
"Contact Number"])
            writer.writerow(data.values())

        messagebox.showinfo("Success", "Registration Successful!")

        for entry in self.entries:
            print(entry.get())

    def clear(self):
        for entry in self.entries:
            entry.delete(0, 'end')

def center_window(window, width=400, height=400):
    screen_width = window.winfo_screenwidth()
    screen_height = window.winfo_screenheight()
    x_coordinate = int((screen_width / 2) - (width / 2))
    y_coordinate = int((screen_height / 2) - (height / 2))
    window.geometry(f"{width}x{height}+{x_coordinate}+{y_coordinate}")
```

C:\Users\TIPQC\Downloads\regis\registration.py

registration.py × main.py ×

```python
from tkinter import *
from tkinter import messagebox
import csv
import os

class RegistrationWindow:
    def __init__(self, win):
        self.label1 = Label(win, text="Account Registration", font=("Impact", 25), bg="Orange", fg="White")
        self.label1.place(x=80, y=10)

        fields = ["First Name:", "Last Name:", "Username:", "Password:", "Email Address:", "Contact Number:"]
        self.entries = []
        y_position = 90

        for i, field in enumerate(fields):
            Label(win, text=field, font=("Arial", 14), anchor='w').place(x=50, y=y_position)
            entry = Entry(win, bd=5)
            entry.place(x=200, y=y_position)
            self.entries.append(entry)
            y_position += 40

        self.submit_btn = Button(win, text="Submit", bd=4, bg="LightGreen", font=("Arial", 14), command=self.submit)
        self.submit_btn.place(x=100, y=y_position + 20)

        self.clear_btn = Button(win, text="Clear", bd=4, bg="LightCoral", font=("Arial", 14), command=self.clear)
        self.clear_btn.place(x=200, y=y_position + 20)

    def submit(self):
        for i, entry in enumerate(self.entries):
            if not entry.get().strip():
                messagebox.showerror("Input Error", f"Make sure nothing is skipped.")
                return

        data = {field: entry.get() for field, entry in zip(["First Name", "Last Name", "Username", "Password", "Email Address", "Contact

        file_exists = os.path.isfile('registrations.csv')
        with open('registrations.csv', mode='a', newline='') as file:
            writer = csv.writer(file)
            if not file_exists:
                writer.writerow(["First Name", "Last Name", "Username", "Password", "Email Address", "Contact Number"])
            writer.writerow(data.values())

        messagebox.showinfo("Success", "Registration Successful!")

        for entry in self.entries:
            print(entry.get())

    def clear(self):
        for entry in self.entries:
            entry.delete(0, 'end')

def center_window(window, width=400, height=400):
    screen_width = window.winfo_screenwidth()
    screen_height = window.winfo_screenheight()
    x_coordinate = int((screen_width / 2) - (width / 2))
    y_coordinate = int((screen_height / 2) - (height / 2))
    window.geometry(f"{width}x{height}+{x_coordinate}+{y_coordinate}")
```

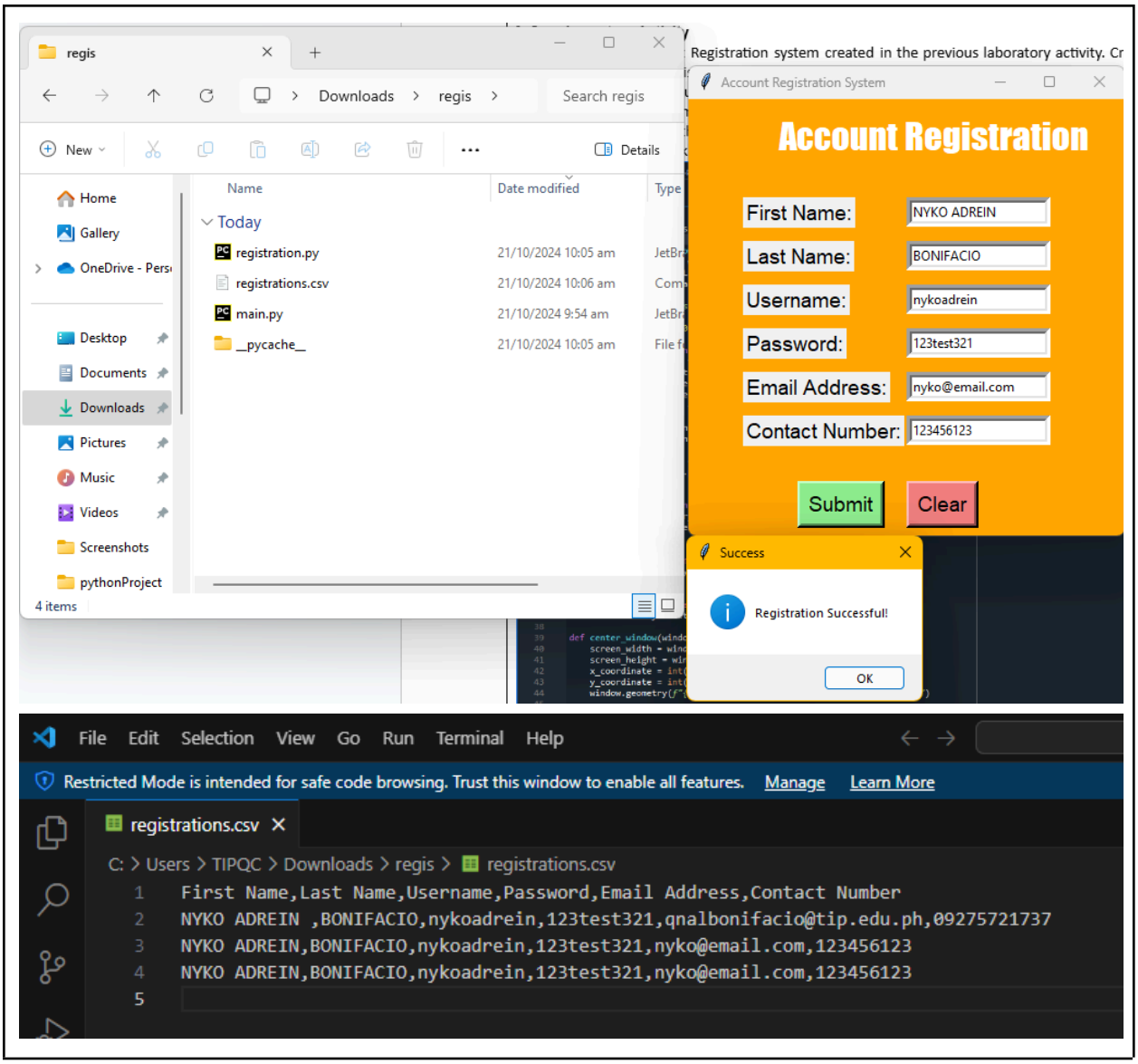
C:\Users\TIPQC\Downloads\regis\main.py

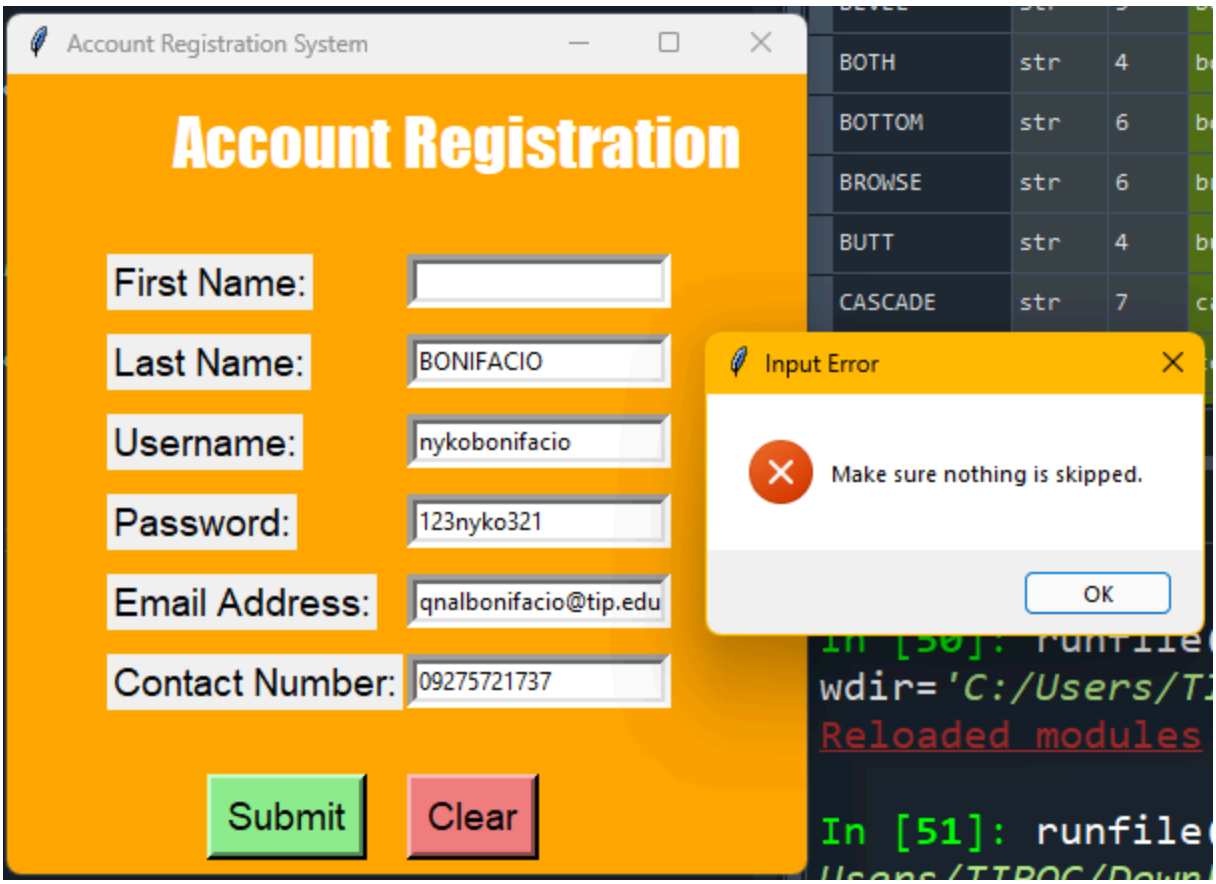
registration.py × main.py* ×

```python
from tkinter import Tk
from registration import RegistrationWindow, center_window

window = Tk()
app = RegistrationWindow(window)

center_window(window, width=400, height=400)
window.title("Account Registration System")
window.configure(bg='Orange')
window.mainloop()
```


regis
Downloads
regis
Search regis
New
Details
Home
Gallery
OneDrive - Pers
Desktop
Documents
Downloads
Pictures
Music
Videos
Screenshots
pythonProject
Name
Date modified
Type
Today
registration.py
21/10/2024 10:05 am
registrations.csv
21/10/2024 10:06 am
main.py
21/10/2024 9:54 am
__pycache__
21/10/2024 10:05 am
4 items
Registration system created in the previous laboratory activity. Cr
Account Registration System
Account Registration
First Name:
NYKO ADREIN
Last Name:
BONIFACIO
Username:
nykoadrein
Password:
123test321
Email Address:
nyko@email.com
Contact Number:
123456123
Submit
Clear
Success
Registration Successful!
OK
File
Edit
Selection
View
Go
Run
Terminal
Help
Restricted Mode is intended for safe code browsing. Trust this window to enable all features.
Manage
Learn More
registrations.csv
C: > Users > TIPQC > Downloads > regis > registrations.csv
1 First Name,Last Name,Username,Password,Email Address,Contact Number
2 NYKO ADREIN ,BONIFACIO,nykoadrein,123test321,qnalbonifacio@tip.edu.ph,09275721737
3 NYKO ADREIN,BONIFACIO,nykoadrein,123test321,nyko@email.com,123456123
4 NYKO ADREIN,BONIFACIO,nykoadrein,123test321,nyko@email.com,123456123
5



Questions

1. What are the other signals available in PyQt5? (give at least 3 and describe each)
   - **clicked()**: This signal is used when a button is clicked. When you click a button this signal gets triggered, and you can run some code.
   - **textChanged()**: This signal is triggered when the text in a text box changes. If you type something new in a text field, this signal lets you know so.
   - **valueChanged()**: This signal is used with sliders or spin boxes. It gets triggered when the value of the slider changes, so you can do something based on the new value.

2. Why do you think that event handling in Python is divided into signals and slots?
   - It's divided because signals and slots make it easier to manage events in a program. A signal tells us that something happened (like a button click), and a slot is where we write what to do when that happens. This way, the code is easier to understand.

3. How can message boxes be used to provide a better User Experience or how can message boxes be used to make a GUI Application more user-friendly?
   - Message boxes can show important information, warnings, or errors to the user. For example, if you try to delete something, a message box can ask if you're sure. This makes the app friendlier because it helps users understand what's happening and prevents mistakes.

4. What is Error-handling and how was it applied in the task performed?
   - Error handling is about managing problems that happen while a program is running. For example, if you try to open a file that doesn't exist, error handling lets you show a message instead of crashing the program. In our tasks, we use this to do trial and error so that we can check our code if it is right or wrong. For example when I'm trying to implement a code where it should check if everything has an input, the first code that I tried to use was wrong but because of the error code I was able to change it and make the code work.

5. What maybe the reasons behind the need to implement error handling?
   - Error handling is important because it helps keep the program running smoothly. It prevents crashes, helps users understand what went wrong, and makes the code or app more reliable. If something goes wrong, it's show a message than to let the app or code to just stop working.

7. Conclusion

- In conclusion, event handling is important for making apps interactive. By understanding events like clicks and key presses, we can create responsive features, such as buttons and menus. This informations can helps us become better programmers and enhances our ability to build enjoyable apps. Overall, event handling is essential in GUI development and opens up creative possibilities.