Laboratory Activity 4 - Introduction to GUI Development using Pycharm	
Jimenez, Christian Joros R.	17/10/2024
CPE21S1	Engr. Rizette Sayo

Main.py

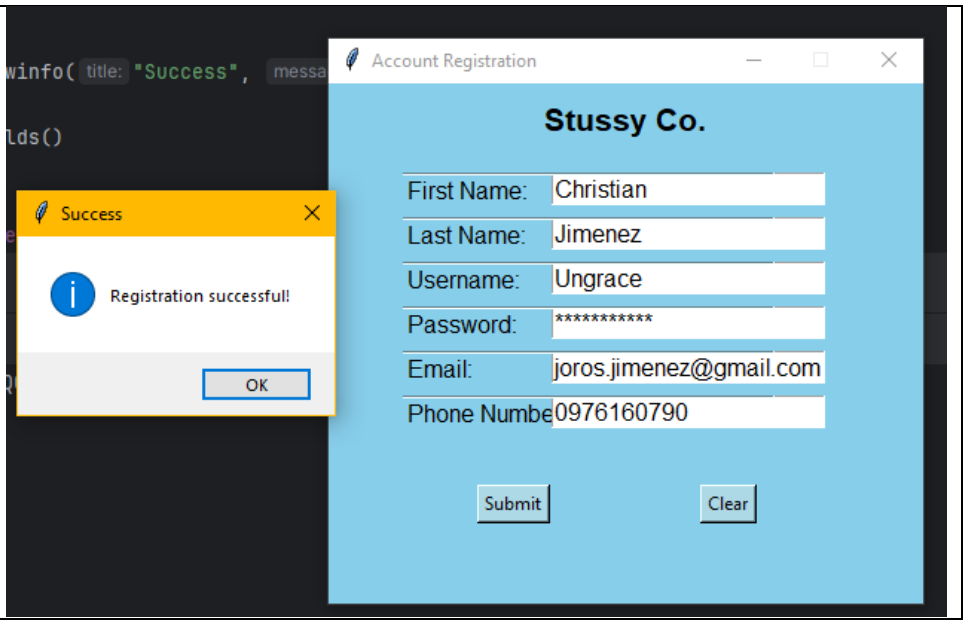
```
import tkinter as tk
from Registration import RegistrationGUI
if __name__ == "__main__":
    root = tk.Tk()
    app = RegistrationGUI(root)
    root.mainloop()


class Registration:
    pass
```

Registration.py

```
import tkinter as tk
from tkinter import ttk, messagebox

class RegistrationGUI:
    def __init__(self, master):
        self.master = master
        master.title("Account Registration")
        master.geometry("400x350")
        master.resizable(False, False)

        screen_width = master.winfo_screenwidth()
        screen_height = master.winfo_screenheight()
        x = (screen_width - master.winfo_width()) // 2
        y = (screen_height - master.winfo_height()) // 2
        master.geometry(f"+{x}+{y}")

        background_color = "SkyBlue"
        master.configure(bg=background_color)

        title_font = ("Arial", 16, "bold")
        self.title_label = tk.Label(master, text="Stussy
Co.", font=title_font, bg=background_color)
        self.title_label.pack(pady=10)

        label_font = ("Helvetica", 12)
        entry_font = ("Helvetica", 12)

        button_bg_color = "lightblue"
        button_fg_color = "black"

        self.first_name_label = tk.Label(master,
text="First Name:", font=label_font, bg=background_color)
        self.first_name_label.place(x=50, y=60)
        self.first_name_entry = tk.Entry(master,
font=entry_font)
```

```
        self.first_name_entry.place(x=150, y=60)

        first_name_frame = tk.Frame(master,
bg=button_bg_color, relief="groove", borderwidth=2)
        first_name_frame.place(x=50, y=60, width=250,
height=2)

        self.last_name_label = tk.Label(master, text="Last
Name:", font=label_font, bg=background_color)
        self.last_name_label.place(x=50, y=90)
        self.last_name_entry = tk.Entry(master,
font=entry_font)
        self.last_name_entry.place(x=150, y=90)

        last_name_frame = tk.Frame(master,
bg=button_bg_color, relief="groove", borderwidth=2)
        last_name_frame.place(x=50, y=90, width=250,
height=2)

        self.username_label = tk.Label(master,
text="Username:", font=label_font, bg=background_color)
        self.username_label.place(x=50, y=120)
        self.username_entry = tk.Entry(master,
font=entry_font)
        self.username_entry.place(x=150, y=120)

        username_frame = tk.Frame(master,
bg=button_bg_color, relief="groove", borderwidth=2)
        username_frame.place(x=50, y=120, width=250,
height=2)

        self.password_label = tk.Label(master,
text="Password:", font=label_font, bg=background_color)
        self.password_label.place(x=50, y=150)
        self.password_entry = tk.Entry(master, show="*",
font=entry_font)
        self.password_entry.place(x=150, y=150)

        password_frame = tk.Frame(master,
bg=button_bg_color, relief="groove", borderwidth=2)
        password_frame.place(x=50, y=150, width=250,
height=2)

        self.email_label = tk.Label(master, text="Email:",
font=label_font, bg=background_color)
        self.email_label.place(x=50, y=180)
        self.email_entry = tk.Entry(master,
font=entry_font)
        self.email_entry.place(x=150, y=180)

        email_frame = tk.Frame(master, bg=button_bg_color,
relief="groove", borderwidth=2)
        email_frame.place(x=50, y=180, width=250,
height=2)

        self.contact_label = tk.Label(master, text="Phone
Number:", font=label_font, bg=background_color)
```

```
        self.contact_label.place(x=50, y=210)
        self.contact_entry = tk.Entry(master,
font=entry_font)
        self.contact_entry.place(x=150, y=210)

        contact_frame = tk.Frame(master,
bg=button_bg_color, relief="groove", borderwidth=2)
        contact_frame.place(x=50, y=210, width=250,
height=2)

        self.submit_button = tk.Button(master,
text="Submit", command=self.submit registration,
bg=button_bg_color, fg=button_fg_color)
        self.submit_button.place(x=100, y=270)

        self.clear_button = tk.Button(master,
text="Clear", command=self.clear_fields,
bg=button_bg_color, fg=button_fg_color)
        self.clear_button.place(x=250, y=270)

    def submit_registration(self):
        first_name = self.first_name_entry.get()
        last_name = self.last_name_entry.get()
        username = self.username_entry.get()
        password = self.password_entry.get()
        email = self.email_entry.get()
        contact = self.contact_entry.get()

        if not all([first_name, last_name, username,
password, email, contact]):
            messagebox.showwarning("Incomplete Data",
"Please fill in all fields.")
            return

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

        self.clear_fields()

    def clear_fields(self):
        self.first_name_entry.delete(0, tk.END)
        self.last_name_entry.delete(0, tk.END)
        self.username_entry.delete(0, tk.END)
        self.password_entry.delete(0, tk.END)
        self.email_entry.delete(0, tk.END)
        self.contact_entry.delete(0, tk.END)

if __name__ == "__main__":
    root = tk.Tk()
    app = RegistrationGUI(root)
    root.mainloop()
```

Output	

Questions

1. What are the common GUI Applications that general end-users such as home users, students, and office employees use? (give at least 3 and describe each)

Web Browsers:

Web browsers are essential for accessing the internet. They allow users to navigate websites, view web pages, download files, and interact with online services. Some examples are Google Chrome, Mozilla Firefox, Microsoft Edge, Safari (for Apple devices). Their common uses are checking email, social media, online shopping, researching information, streaming videos, accessing online banking, etc.

Word Processors:

Word processors are software applications designed for creating and editing documents. They provide features like formatting text, inserting images, creating tables, and checking spelling and grammar. Some examples are Microsoft Word, Google Docs, LibreOffice Writer, Apple Pages. Their common Uses: Writing essays, reports, letters, resumes, creating presentations, drafting notes, etc.

Email Clients:

Email clients enable users to send and receive emails. They offer features like composing and replying to messages, organizing emails into folders, attaching files, and managing contacts. Some examples are icrosoft Outlook, Gmail, Apple Mail,

Thunderbird. Their common uses are communicating with others, sending and receiving important documents, scheduling meetings, managing contacts, etc.

2. Based from your answer in question 1, why do you think home users, students, and office employees use those GUI programs?

GUI applications such as word processors, email clients, and web browsers are indispensable for everyday work for office workers, students, and home users alike. Regardless of one's level of technical skill, these programs' clear and user-friendly interfaces make technology accessible to all. These applications are used by home users to stay in touch, handle money, and have fun. Students use them for group projects, research projects, and homework completion. They are essential to business operations, document creation, and communication for office workers. GUI programs' visual and functional qualities enable people with a variety of backgrounds to easily interact with technology and complete a wide range of tasks.

3. How does Pycharm help developers in making GUI applications, what would be the difference if developers made GUI programs without GUI Frameworks such as Pycharm or Tkinter?

Python GUI application development is made a lot easier with PyCharm's code completion, visual editors, integrated debugging, and refactoring tools. It gives developers higher-level tools and abstractions to make layout management, event handling, and widget creation easier. In comparison, creating GUIs directly without the use of a framework calls for possibly lower-level code, less visual feedback, and more manual coding. With the help of frameworks like PyCharm, GUI development is now more accessible and developers can produce functional and aesthetically pleasing user interfaces with less effort. These frameworks also offer efficiency and support for advanced features.

4. What are the different platforms a GUI program may be created and deployed on? (Three is required then state why might a program be created on that specific platform)

There are several platforms on which GUI programs can be developed and implemented, each with its own set of benefits. Windows is frequently the main target for wide reach due to its large user base and recognizable interface. Users who appreciate a unified and aesthetically pleasing experience are drawn to macOS, which is renowned for its aesthetics and compatibility with Apple products. Web applications are perfect for reaching a global audience because they are cross-platform compatible, easy to distribute, and accessible from any device with an internet connection. The choice of platform for GUI development depends on factors like target audience, application requirements, and the developer's resources and preferences.

5. What is the purpose of **app = QApplication(sys.argv)**, **ex = App()**, and **sys.exit(app.exec_())**?

QApplication(sys.argv) = app

The central component of PyQt's GUI system is called QApplication. It generates the application object that controls the main window, the event loop, and every other GUI component.. The list of command-line arguments that your Python script is given is called sys.argv.. This is necessary for QApplication to manage any potential arguments that are passed to the application (e.g., whether your application needs to launch in a particular mode).
To put it briefly, this line initializes the PyQt application and sets it up to handle events and user input.

x = App()

Most likely, this is referring to a class that you have defined in your code to represent either the application's main window or a particular widget.

ex = App(): This generates an instance of the App class (or any other class you are working with), i.e., it creates and configures the GUI elements themselves.
To put it simply: This line launches an instance of the main GUI component.

app.exec_();sys.exit():

The PyQt event loop is started by using app.exec_(). The event loop is necessary because it continuously scans the system for events (keystrokes, mouse clicks, etc.) and responds to them.
Sys.exit(): This provides a graceful way to end a Python script.
This line essentially instructs the application to start its GUI and to end gracefully when the GUI is closed.

Conclusion:

In conclusion, I was able to create my own account registration program using pycharm. I was able to familiarize myself witht eh pycharm framework and processes in creating GUI systems, I created two separate files for the program itself, I learned how to import another file in the same directory, making it clean and organized, and also the GUI code reusable for other projects. I also familiarized myself with the tkinter module which is repsonsibile for making GUI in python.