EXPERIMENT – 5:

AIM: To study and implement program on creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes.

THEORY:

Tkinter is a Python Package for creating GUI applications. Python has a lot of GUI frameworks, but Tkinter is the only framework that's built into the Python standard library.

Tkinter has several strengths; it's cross-platform, so the same code works on Windows, macOS, and Linux.

Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python, especially for applications where a modern shine is unnecessary, and the top priority is to build something functional and cross-platform quickly.

- **Use Cases of Tkinter:**

1. Creating windows and dialog boxes: Tkinter can be used to create windows and dialog boxes that allow users to interact with your program. These can be used to display information, gather input, or present options to the user.

To create a window or dialog box, you can use the *Tk()* function to create a root window, and then use functions like *Label*, *Button*, and *Entry* to add widgets to the window.

2. Building a GUI for a desktop application: Tkinter can be used to create the interface for a desktop application, including buttons, menus, and other interactive elements.

To build a GUI for a desktop application, you can use functions like *Menu*, *Checkbutton*, and *RadioButton* to create menus and interactive elements and use layout managers like *pack* and *grid* to arrange the widgets on the window.

3. Adding a GUI to a command-line program: Tkinter can be used to add a GUI to a command-line program, making it easier for users to interact with the program and input arguments.

To add a GUI to a command-line program, you can use functions like *Entry* and *Button* to create input fields and buttons, and use event handlers like *command* and *bind* to handle user input.

4. Creating custom widgets: Tkinter includes a variety of built-in widgets, such as buttons, labels, and text boxes, but it also allows you to create your own custom widgets.

To create a custom widget, you can define a class that inherits from the *Widget* class and overrides its methods to define the behavior and appearance of the widget.

5. Prototyping a GUI: Tkinter can be used to quickly prototype a GUI, allowing you to test and iterate on different design ideas before committing to a final implementation.

To prototype a GUI with Tkinter, you can use the *Tk()* function to create a root window, and then use functions like *Label*, *Button*, and *Entry* to add widgets to the window and test different layouts and design ideas.

➢ Getting Started with Tkinter:

1. Import tkinter package and all of its modules.

2. Create a root window. Give the root window a title(using **title()**) and dimension(using **geometry()**). All other widgets will be inside the root window.

3. Use **mainloop()** to call the endless loop of the window. If you forget to call this nothing will appear to the user. The window will wait for any user interaction till we close it.

➢ Some of the common widgets used in Tkinter are :

- **Frame**: serves as a holding area for other widgets and serves as a container.
- **Text:** It enables us to display and alter text in a variety of styles and offers a prepared text display.
- **Label:** Used to display text and images, but we are unable to interact with it.
- **Button:** Often used add buttons and we may add functions and methods to it.
- **Entry:** One-line string text can be entered into this widget.
- **Labelframe:** For intricate window layouts, this widget serves as a separator or container.
- **Listbox:** It just has text elements, all of which are the same colour and font.
- **Scrollbar:** This gives a sliding controller.
- **Canvas:** Custom widgets can be implemented using the canvas widget.
- **Scale:** This widget offers graphical slider items that let us choose different scale values.

- **Radiabutton**: Use a radio button to carry out one of several choices.
- **Checkbox**: Use a checkbox to implement on-off choices.
- **Listbox:** It just has text elements, all of which are the same colour and font.

➢ PROGRAM:

```
from tkinter import *
from tkinter import messagebox

def register():
   name = en1.get()
   messagebox.showinfo("Registration Successful", f"Registration
successful for {name}")

base = Tk()
base.geometry("500x500")
base.title("Tkinter Form S21 (2201084)")

lb1 = Label(base, text="Enter First Name", font=("Times New
Roman",12))
lb1.place(x=20, y=60)
en1 = Entry(base)
en1.place(x=200, y=60)

lb2 = Label(base, text="Enter Last Name",  font=("Times New
Roman",12))
lb2.place(x=20, y=100)
en2 = Entry(base)
en2.place(x=200, y=100)

lb3 = Label(base, text="Enter Email", width=10, font=("Times New
Roman",12))
lb3.place(x=10, y=140)
en3 = Entry(base)
en3.place(x=200, y=140)

lb4 = Label(base, text="Contact Number", font=("Times New
Roman",12))
lb4.place(x=15, y=180)
en4 = Entry(base)
```

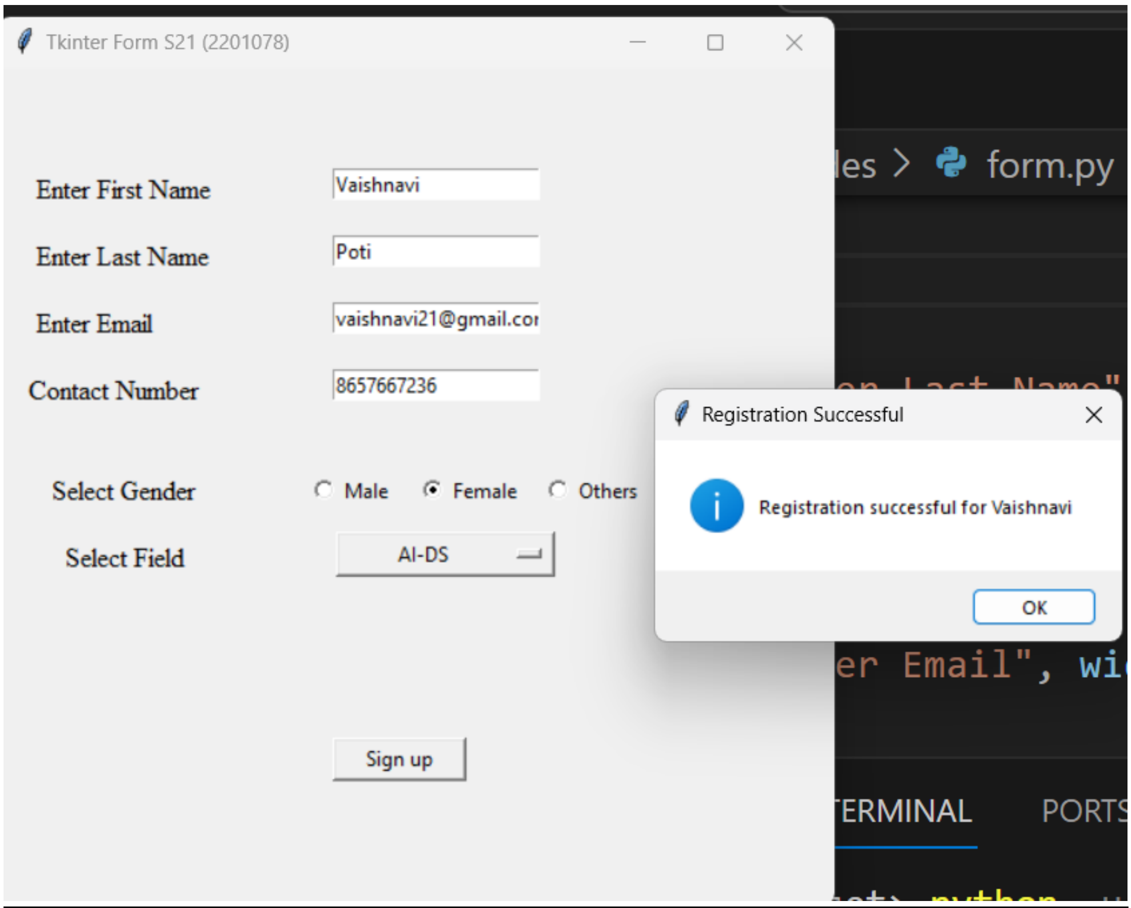
```
en4.place(x=200, y=180)

lb5 = Label(base, text="Select Gender", width=15, font=("Times New
Roman",12))
lb5.place(x=5, y=240)
vars = IntVar()
Radiobutton(base, text="Male", padx=5, variable=vars,
value=1).place(x=180, y=240)
Radiobutton(base, text="Female", padx=10, variable=vars,
value=2).place(x=240, y=240)
Radiobutton(base, text="Others", padx=15, variable=vars,
value=3).place(x=310, y=240)

list_of_fields = ("AI-DS", "IT", "Comps", "EXTC")
cv = StringVar()
drplist = OptionMenu(base, cv, *list_of_fields)
drplist.config(width=15)
cv.set("Select Field")
lb2 = Label(base, text="Select Field", width=13, font=("Times New
Roman",12))
lb2.place(x=14, y=280)
drplist.place(x=200, y=275)

Button(base, text="Sign up", width=10, command=register).place(x=200,
y=400)
base.mainloop()
```

- **OUTPUT:**



Enter First Name: Vaishnavi
Enter Last Name: Poti
Enter Email: vaishnavi21@gmail.cor
Contact Number: 8657667236
Select Gender: Male [✓ Female] Others
Select Field: AI-DS

Registration successful for Vaishnavi

- **CONCLUSION:** Hence, we have successfully implemented program on creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes; LO 1.