

**St. Francis Institute of Technology,
Mumbai-400 103**

A.Y. 2023-24
Class: SE-ITA/ITB,
Semester: IV Subject:
Python Lab.

Experiment –12: GUI and Database Connectivity using Python

1. Aim: To implement a Graphical User Interface in Python using tkinter and Database using Sqlite3
2. Objectives: After study of this experiment, the student will be able to implement GUI using tkinter including windows, frames, canvas and widgets.
3. Outcomes: Students shall be able to create GUI for application and connect it to a database (LO- 404.3).
4. **Prerequisite:** Knowledge of Python basics.
5. **Requirements:** Personal Computer (PC), Windows /Linux Operating System, Python IDE.

6. Pre-Experiment Exercise:

Theory:

The tkinter package (Tk interface) is the standard Python interface to the Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, as well as on Windows systems. Python offers tkinter module to create graphics programs. It enables use of the classes TK module of TCL/TK language. A window is where all the GUI is visible a window is an object of class Tk. The frames and canvas or a widget is added to the window using a pack(), grid() or place() method.

pack(): It organizes the widgets in the block.

grid(): It organizes the widgets in table-like structure.

place(): It's used to place the widgets at a specific position you want.

A widget is a GUI component that is displayed on the screen and can perform a task as given by the user. Different widgets in Python are: Text, Label, Dropdown, Checkbutton, Radiobutton, Entry, Listbox, Menu, Spinbox, ScrollBar.

To use SQLite3 in Python, import the sqlite3 module and then create a connection object which will connect us to the database and will let us execute the SQL statements. To execute SQLite statements in Python, you need a cursor object, create it using the cursor() method. Use the cursor object to call the execute() method to execute any SQL queries for creating a table, adding entries in the table.

7. Laboratory Exercise:

A. Procedure:

- i. Open IDE for python programming
- ii. Open new python file from menu file-new
- iii. Type python code with proper syntax
- iv. Save file with .py extension
- v. Execute and observe the output

B. Program code with comments:

Write and execute your program code to achieve the given aim and attach it with your own comments with neat indentation.

8. Post-Experiments Exercise:

A. Extended theory:

- i. Explain the use of a canvas and create a shape using the same.

ii. Explain the syntax and explore all the different widgets that can be used in Tkinter.

iii. Show creation of image buttons and use of MessageBox in tkinter Python GUI.

B. Results/Observations/Program output:

Present the program input/output results and comment on the same

C. Questions/Programs:

i. Create a GUI with frame and widgets: First name and Last name (Text), Gender (Radiobutton), Subject (dropdown menu)

ii. Add 3 buttons the GUI window created in i. To display the given input, reset and exit the window.

ii. Connect the form to a database and enter values from the form into the database.

D. Conclusion:

1. Write what was performed in the experiment/program.

2. What is the significance of experiment/program?

3. Mention a few applications of what was studied.

E. References:

[1] James Payne, “Beginning Python: Using Python 2.6 and Python 3.1”, Wrox Publication

[2] <https://docs.python.org/3/library/tkinter.html#tkinter-modules>

[3] <https://docs.python.org/3/library/sqlite3.html>

1. Create a GUI with frame and widgets: First name and Last name (Text), Gender (Radiobutton), Subject (dropdown menu)

```
import tkinter as tk

from tkinter import ttk

# Create the main window

root = tk.Tk()

root.title("Form")

# Create a frame

frame = ttk.Frame(root, padding="10")

frame.pack()

# Create 'First name' and 'Last name' labels and text entry boxes

first_name_label = ttk.Label(frame, text="First Name:",
font=('Helvetica', 10))

first_name_label.grid(row=0, column=0, sticky=tk.W, pady=2)

first_name_entry = ttk.Entry(frame)

first_name_entry.grid(row=0, column=1, pady=2)
```

```
last_name_label = ttk.Label(frame, text="Last Name:", font=('Helvetica',
10))

last_name_label.grid(row=1, column=0, sticky=tk.W, pady=2)

last_name_entry = ttk.Entry(frame)

last_name_entry.grid(row=1, column=1, pady=2)


# Create 'Gender' radio buttons

gender_label = ttk.Label(frame, text="Gender:", font=('Helvetica', 10))

gender_label.grid(row=2, column=0, sticky=tk.W, pady=2)

gender = tk.StringVar()

male_radio = ttk.Radiobutton(frame, text="Male", variable=gender,
value="Male")

male_radio.grid(row=2, column=1, sticky=tk.W, pady=2)

female_radio = ttk.Radiobutton(frame, text="Female", variable=gender,
value="Female")

female_radio.grid(row=3, column=1, sticky=tk.W, pady=2)


# Create 'Subject' dropdown menu

subject_label = ttk.Label(frame, text="Subject:", font=('Helvetica',
10))

subject_label.grid(row=4, column=0, sticky=tk.W, pady=2)

subject = ttk.Combobox(frame, values=["Automata Theory", "EM-4",
```

```

"COA","UNIX"]])

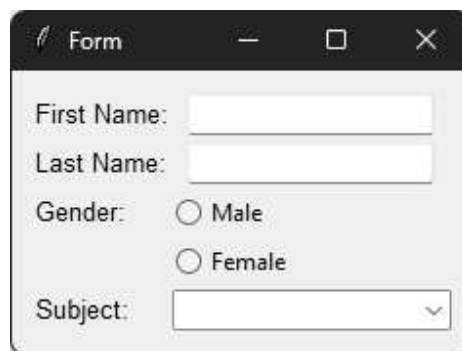
subject.grid(row=4, column=1, pady=2)

# Run the main loop

root.mainloop()

```

OUTPUT:



2. Add 3 buttons the GUI window created in 1. To display the given input, reset and exit the window.

Added CodeBlock:

```

def display_input():

    messagebox.showinfo("Input", f"First Name:
{first_name_entry.get()}\nLast Name: {last_name_entry.get()}\nGender:
{gender.get()}\nSubject: {subject.get()}")

# Function to reset the form

```

```
def reset_form():

    first_name_entry.delete(0, tk.END)

    last_name_entry.delete(0, tk.END)

    gender.set('')

    subject.set('')


# Create 'Display', 'Reset', and 'Exit' buttons

display_button = ttk.Button(frame, text="Submit/Display",
command=display_input)

display_button.grid(row=5, column=0, pady=1)


reset_button = ttk.Button(frame, text="Reset", command=reset_form)

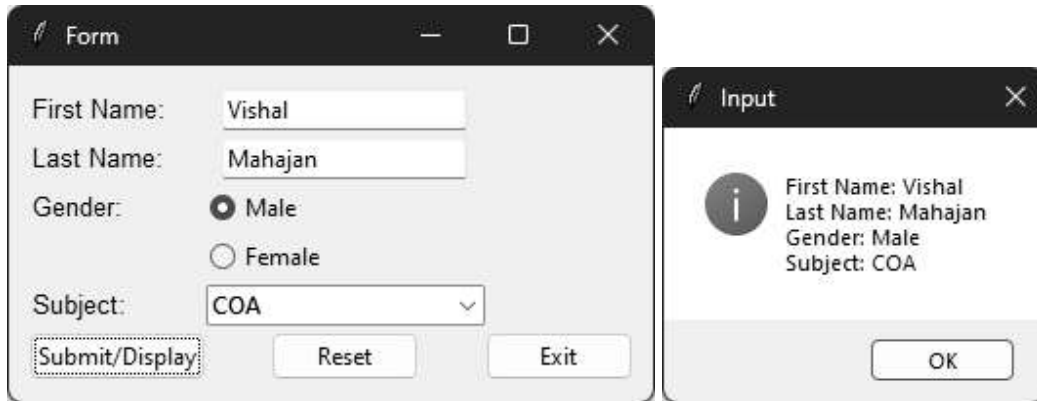
reset_button.grid(row=5, column=1, pady=1)


exit_button = ttk.Button(frame, text="Exit", command=root.destroy)

exit_button.grid(row=5, column=2, pady=1)
```


Output:

1) Display



The image shows two windows from a Java Swing application. The 'Form' window on the left contains the following elements:

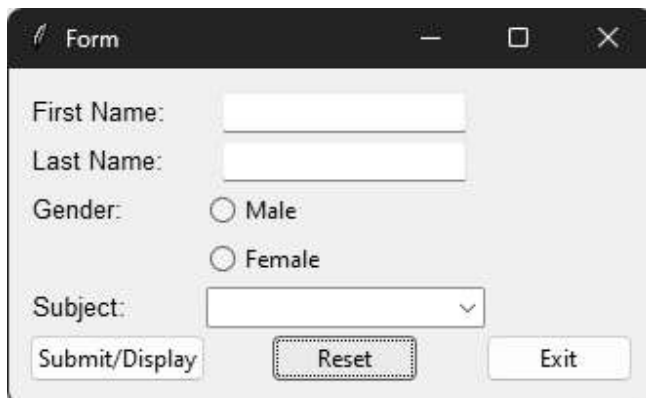
- First Name:
- Last Name:
- Gender: ☒ Male, ☐ Female
- Subject:
- Buttons: Submit/Display, Reset, Exit

The 'Input' window on the right displays the data entered in the form:

First Name: Vishal
Last Name: Mahajan
Gender: Male
Subject: COA

It has an 'OK' button at the bottom.

2) Reset



The image shows the 'Form' window after the 'Reset' button has been clicked. All input fields are now empty:

- First Name:
- Last Name:
- Gender: ☐ Male, ☐ Female
- Subject:
- Buttons: Submit/Display, Reset, Exit

3. Connect the form to a database and enter values from the form into the database.

Added Code:

```
# Connect to the SQLite database (or create it if it doesn't exist)

conn = sqlite3.connect('form_data.db')

c = conn.cursor()


# Create a new table (if it doesn't exist)

c.execute('''

    CREATE TABLE IF NOT EXISTS form_data (

        first_name TEXT,

        last_name TEXT,

        gender TEXT,

        subject TEXT

    )

''')


# Function to submit the form

def submit_form():

    # Insert the form data into the database
```

```

c.execute('''

    INSERT INTO form_data VALUES (?, ?, ?, ?)

    ''', (first_name_entry.get(), last_name_entry.get(), gender.get(),
subject.get()))

# Commit the changes and close the connection

conn.commit()


# Display the input

display_input()


# Function to display the input

def display_input():

    messagebox.showinfo("Input", f"First Name: {first_name_entry.get()}\nLast
Name: {last_name_entry.get()}\nGender: {gender.get()}\nSubject:
{subject.get()}")

```

And Updated the Submit/Display Button to Submit

```

submit_button = ttk.Button(frame, text="Submit/Display", command=submit_form)

submit_button.grid(row=5, column=0, pady=1)

```

Output:

Form

First Name: Vishal

Last Name: Mahajan

Gender: ☒ Male ☐ Female

Subject: Automata Theory

Submit/Display

Reset

Exit

Input

i

First Name: Vishal
Last Name: Mahajan
Gender: Male
Subject: Automata Theory

OK

DB Browser for SQLite - F:\College Stuff\Vishal Mahajan SE IT SEM 4\form_data.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Attach Database

Database Structure Browse Data Edit Pragmas Edit Database Cell

Table: form_data

	first_name	last_name	gender	subject
1	Vishal	Mahajan	Male	Automata Theory

Type of data currently in cell: Text / Numeric
6 character(s)

Remote

Identity Select an identity to connect

DBHub.io Local Current Database

Name	Last modified	Size
------	---------------	------

SQL Log Plot DB Schema Remote

UTF-8