Exercise 3 **Date: 01-02-25**

Develop and compare CLI, GUI, and Voice User Interfaces (VUI) for the same task and assess user satisfaction using Python (Tkinter for GUI, Speech Recognition for VUI), Terminal

AIM:

The aim is to develop and compare Command Line Interface (CLI), Graphical User Interface (GUI), and Voice User Interface (VUI) for the same task, and assess user satisfaction using Python (with Tkinter for GUI and Speech Recognition for VUI) and Terminal.

PROCEDURE:

i) CLI (Command Line Interface)

CLI implementation where users can add, view, and remove tasks using the terminal.

```
 tasks = []
def add_task(task):
    tasks.append(task)
    print(f"Task '{task}' added.")
 def
view_tasks():
    if tasks:
        print("Your tasks:") for idx, task
        in enumerate(tasks, 1):
            print(f"{idx}. {task}")
    else:
        print("No tasks to show.")
 def
remove_task(task_number):
    if 0 < task_number <= len(tasks):
```

```
        removed_task = tasks.pop(task_number - 1)
        print(f"Task '{removed_task}' removed.")
    else:
        print("Invalid task number.")
 def
main():
    while True: print("\nOptions: 1.Add Task 2.View Tasks
        3.Remove
Task 4.Exit") choice = input("Enter your
        choice: ")
 if choice == '1.': task = input("Enter
        task: ") add_task(task)
        elif choice == '2.':
            view_tasks()
        elif choice == '3':
            task_number = int(input("Enter task number to
remove: "))
            remove_task(task_number)
        elif choice == '4':
            print("Exiting...")
            break
        else:
            print("Invalid choice. Please try again.")
 if  name  == " main ":
   main()
```

ii) GUI (Graphical User Interface)

Tkinter to create a simple GUI for our To-Do List application.

```
 import tkinter as tk
from tkinter import messagebox

tasks = []

def  add_task():  task  =
    task_entry.get()     if
    task:
```

```
        tasks.append(task)
        task_entry.delete(0, tk.END)
        update_task_list()
    else: messagebox.showwarning("Warning", "Task cannot
        be
empty")
 def        update_task_list():
task_list.delete(0, tk.END) for
task in tasks:
        task_list.insert(tk.END, task)
 def
remove_task():
    selected_task_index = task_list.curselection()
    if selected_task_index:
        task_list.delete(selected_task_index)
        tasks.pop(selected_task_index[0])
 app      =      tk.Tk()
app.title("To-Do List")

task_entry = tk.Entry(app, width=40)
task_entry.pack(pady=10)
add_button = tk.Button(app, text="Add
Task", command=add_task)
add_button.pack(pady=5)

remove_button = tk.Button(app, text="Remove
Task", command=remove_task)
remove_button.pack(pady=5)

task_list = tk.Listbox(app, width=40, height=10)
task_list.pack(pady=10)

app.mainloop()
```

iii) VUI (Voice User Interface)

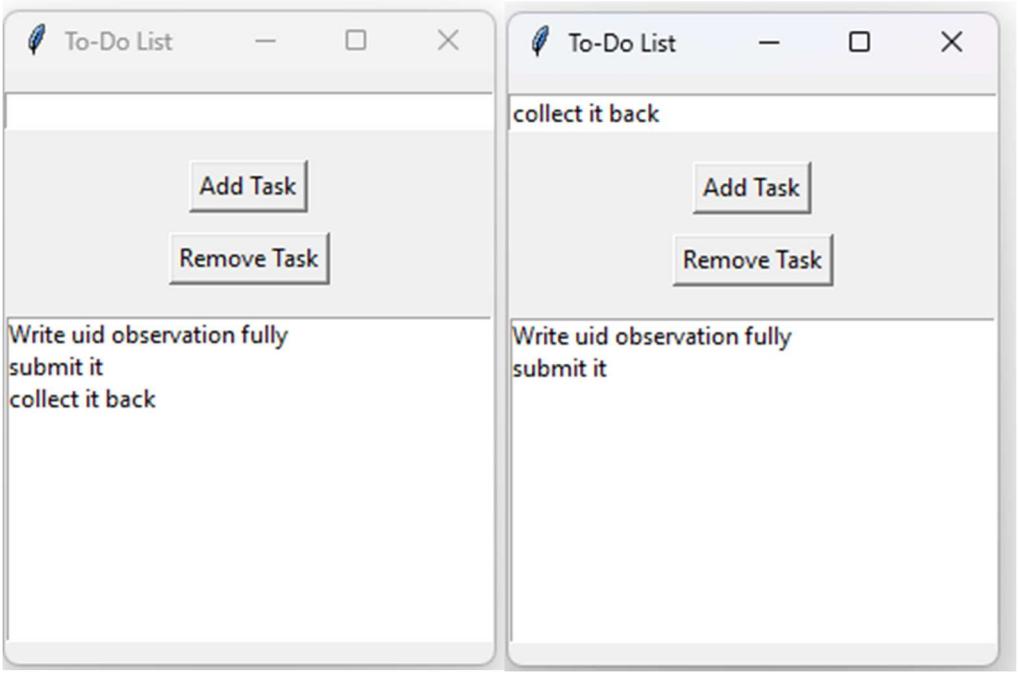
speech_recognition library for voice input and the pyttsx3 library for text-to-speech output. Make sure you have these libraries installed (pip install SpeechRecognition pyttsx3).

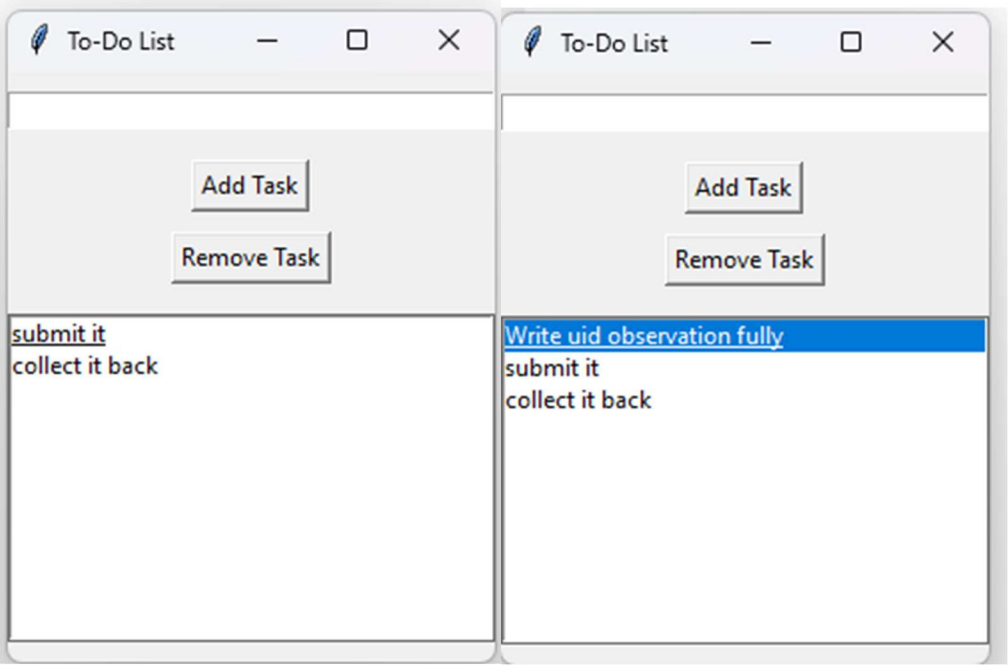
```
 import speech_recognition as sr
import pyttsx3
 tasks = [] recognizer =
sr.Recognizer() engine =
pyttsx3.init()
 def
add_task(task):
    tasks.append(task)
    engine.say(f"Task {task} added")
    engine.runAndWait()
 def
view_tasks():
    if tasks:
        engine.say("Your tasks are")
        for task in tasks:
            engine.say(task)
    else:
        engine.say("No tasks to show")
    engine.runAndWait()
 def
remove_task(task_number):
    if 0 < task_number <= len(tasks):
        removed_task = tasks.pop(task_number - 1)
        engine.say(f"Task {removed_task} removed")
    else:
        engine.say("Invalid task number")
    engine.runAndWait()
 def
recognize_speech():
```

```
    with sr.Microphone() as source:
        print("Listening...") audio =
        recognizer.listen(source) try:
            command = recognizer.recognize_google(audio)
            return command
        except sr.UnknownValueError: engine.say("Sorry, I
            did not understand that") engine.runAndWait()
            return None
 def
main():
while
True:
engine.say
("Options:
add task,
view
tasks,
remove
task, or exit")
        engine.runAndWait()
 command = recognize_speech() if not
        command:
            continue
 if "add task" in command: engine.say("What
        is the task?") engine.runAndWait()
        task = recognize_speech() if task:
                add_task(task)
        elif "view tasks" in command:
            view_tasks()
        elif "remove task" in command:
            engine.say("Which task number to remove?")
            engine.runAndWait()      task_number      =
            recognize_speech()      if      task_number:
            remove_task(int(task_number))
        elif "exit" in command:
            engine.say("Exiting...")
            engine.runAndWait()
            break
        else:
```

```
            engine.say("Invalid option. Please try again.")
            engine.runAndWait()
if __name__ == "__main__":
    main()
```

```
Options: 1.Add Task 2.View Tasks 3.Remove Task 4.Exit
Enter your choice: 1
Enter task: write uid observation
Task 'write uid observation' added.

Options: 1.Add Task 2.View Tasks 3.Remove Task 4.Exit
Enter your choice: 1
Enter task: submit the observation note
Task 'submit the observation note' added.

Options: 1.Add Task 2.View Tasks 3.Remove Task 4.Exit
Enter your choice: 2
Your tasks:
1. write uid observation
2. submit the observation note

Options: 1.Add Task 2.View Tasks 3.Remove Task 4.Exit
Enter your choice: 3
Enter task number to remove: 2
Task 'submit the observation note' removed.

Options: 1.Add Task 2.View Tasks 3.Remove Task 4.Exit
Enter your choice: 4
Exiting...
```





Result

GUI gave the best user satisfaction, CLI was fastest for experts, and VUI was accessible but less accurate.