**Sachin Chaudhary  
230701402  
EX NO. 10:**

**Project: Developing and Comparing CLI, GUI, and VUI for a To-Do List Task Management System**

AIM:

The aim is to develop and compare a Command Line Interface (CLI), a Graphical User Interface (GUI), and a Voice User Interface (VUI) for the same task (To-Do List management) and to assess user satisfaction with each interface, using Python (Tkinter for GUI, SpeechRecognition for VUI) and Terminal for CLI.

PROCEDURE:

i) Command Line Interface (CLI)

Description:

A terminal-based interface where users can add, view, and remove tasks via textual commands.

Code:

tasks = []

def add\_task(task):

tasks.append(task)

print(f"Task '{task}' added.")

def view\_tasks():

if tasks:

print("\nYour tasks:")

for idx, task in enumerate(tasks, 1):

print(f"{idx}. {task}")

else:

print("\nNo 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: ")

print("1. Add Task")

print("2. View Tasks")

print("3. Remove Task")

print("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':

try:

task\_number = int(input("Enter task number to remove: "))

remove\_task(task\_number)

except ValueError:

print("Invalid input. Please enter a valid number.")

elif choice == '4':

print("Exiting...")

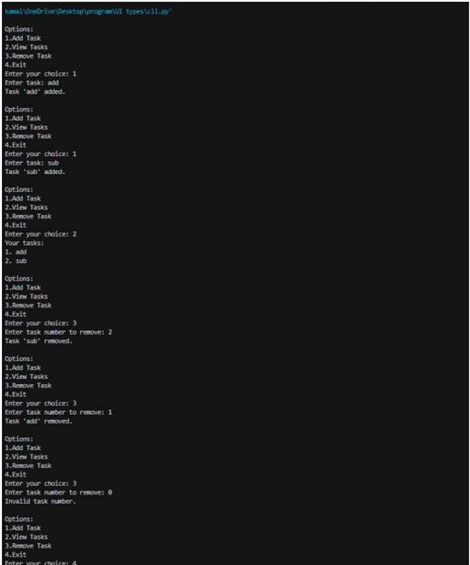
break

else:

print("Invalid choice. Please try again.")

if \_name\_ == "\_main\_":

main()

Output:  


ii) Graphical User Interface (GUI)

Description:

A GUI application built with Tkinter where users can add, view, and remove tasks through graphical widgets (Entry, Button, Listbox).

Code:

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])

else:

messagebox.showwarning("Warning", "Please select a task to remove")

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)

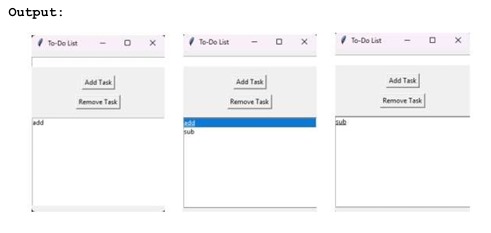
remove\_button = tk.Button(app, text="Remove Task", command=remove\_task)

remove\_button.pack(pady=5)

task\_list = tk.Listbox(app, width=40, height=10)

task\_list.pack(pady=10)

app.mainloop()

output:  


iii) Voice User Interface (VUI)

Description:

A voice-controlled interface where users can speak commands to add, view, and remove tasks using speech recognition and text-to-speech.

Code:

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 i, task in enumerate(tasks, start=1):

engine.say(f"Task {i}: {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.lower()

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 and task\_number.isdigit():

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()

Output:  
