EXPERIMENT – 3

Study and Implementation of CLI, GUI and VUI

CLI (Command Line Interface)

A Command Line Interface (CLI) is a text-based interface where users interact with a system by typing commands in the terminal.
It is fast, lightweight, and mainly used by technical users.

IMPLEMENTATION CODE :

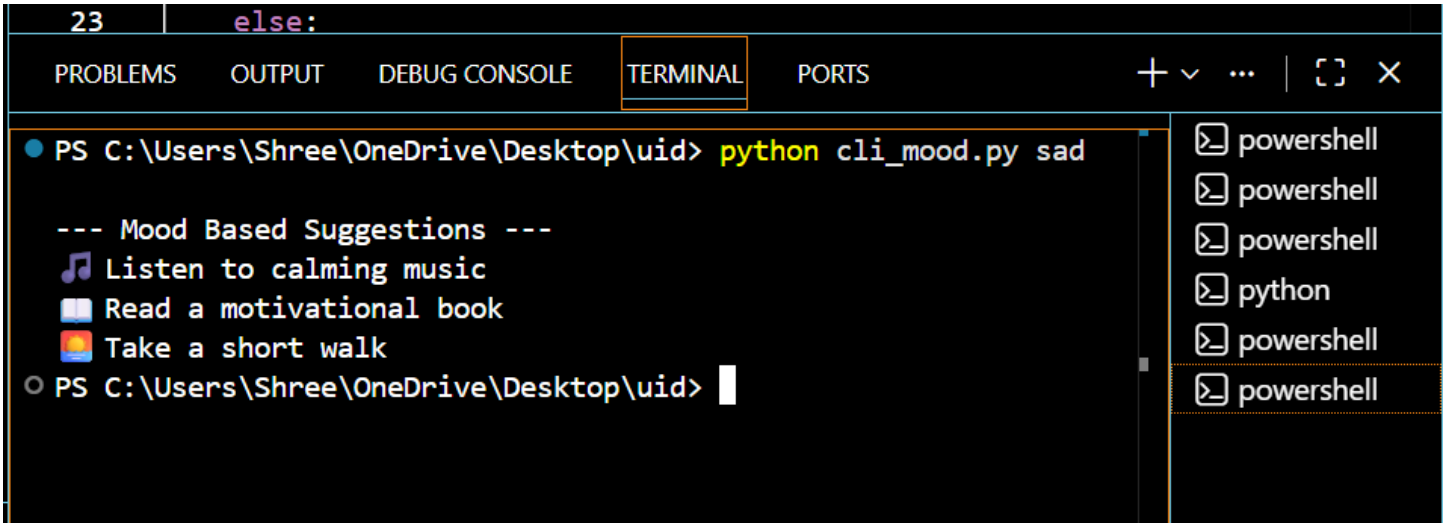
```
import sys

def suggest_activity(mood):

    mood = mood.lower()

print("\n--- Mood Based Suggestions ---")

if mood == "happy":

        print("🎵 Listen to energetic music")

        print("📸 Go out and click some photos")

        print("🍦 Treat yourself with ice cream")

    elif mood == "sad":

        print("🎵 Listen to calming music")

        print("📖 Read a motivational book")

        print("🌅 Take a short walk")

    elif mood == "stressed":

        print("🧘 Try meditation for 5 minutes")

        print("🎧 Listen to relaxing sounds")

        print("☕ Drink some tea")

    else:

        print("Mood not recognized. Try: happy, sad, stressed")
```

```
if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: python cli_mood.py <mood>")
    else:
        suggest_activity(sys.argv[1])
```

How It Works:

1. The program imports the sys module to access command-line arguments.
2. The user provides mood as input through the terminal.
3. The program checks the mood using conditional statements.
4. Based on the mood, it prints suitable suggestions.
5. If the mood is not recognized, it displays an error message.
6. The output is displayed directly in the terminal window.

OUTPUT :



GUI (Graphical User Interface)

A Graphical User Interface (GUI) allows users to interact with the system using visual elements like buttons, labels, and windows.
It is user-friendly and suitable for beginners.

IMPLEMENTATION CODE:

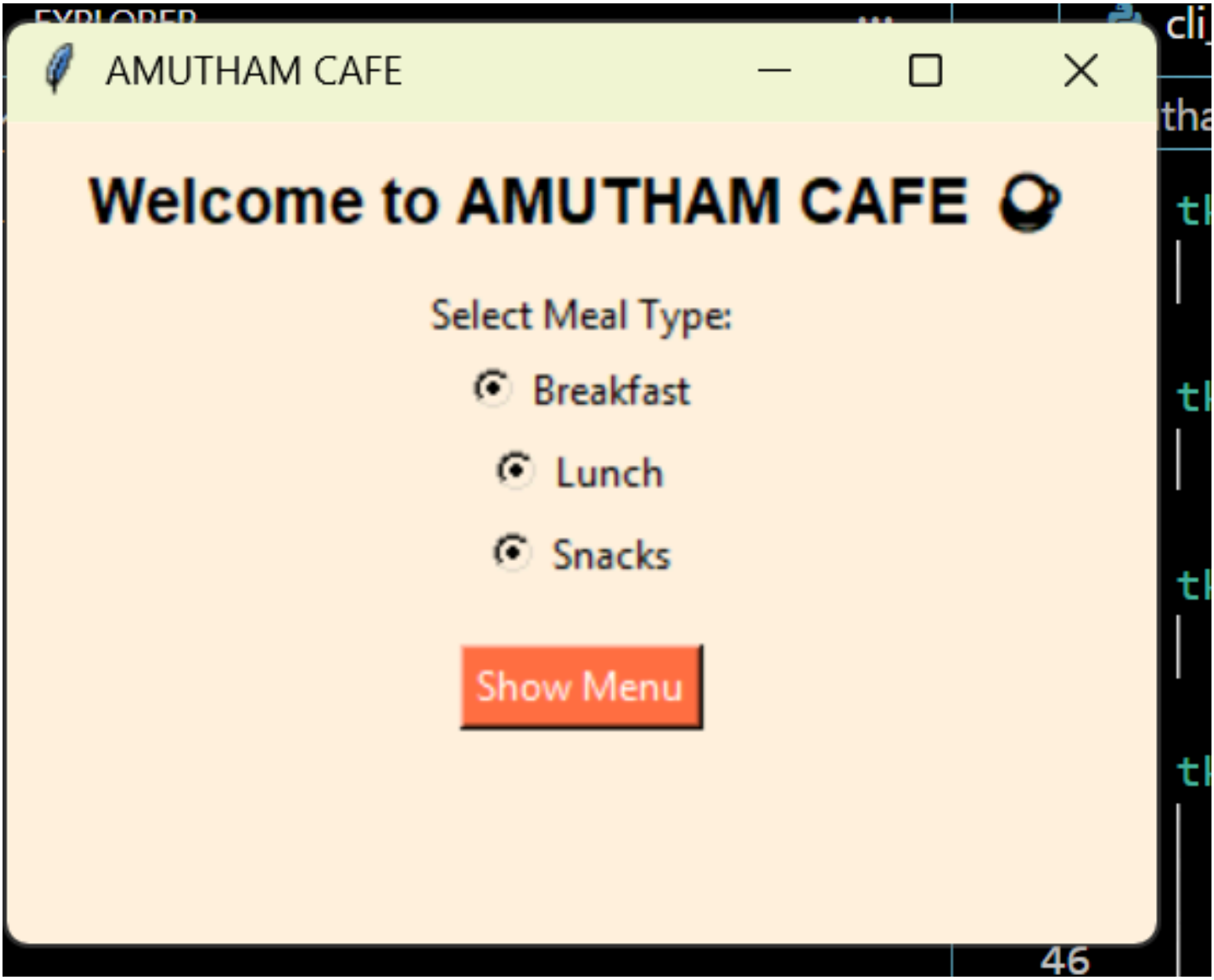
```
import tkinter as tk

from tkinter import messagebox

def show_menu():
```

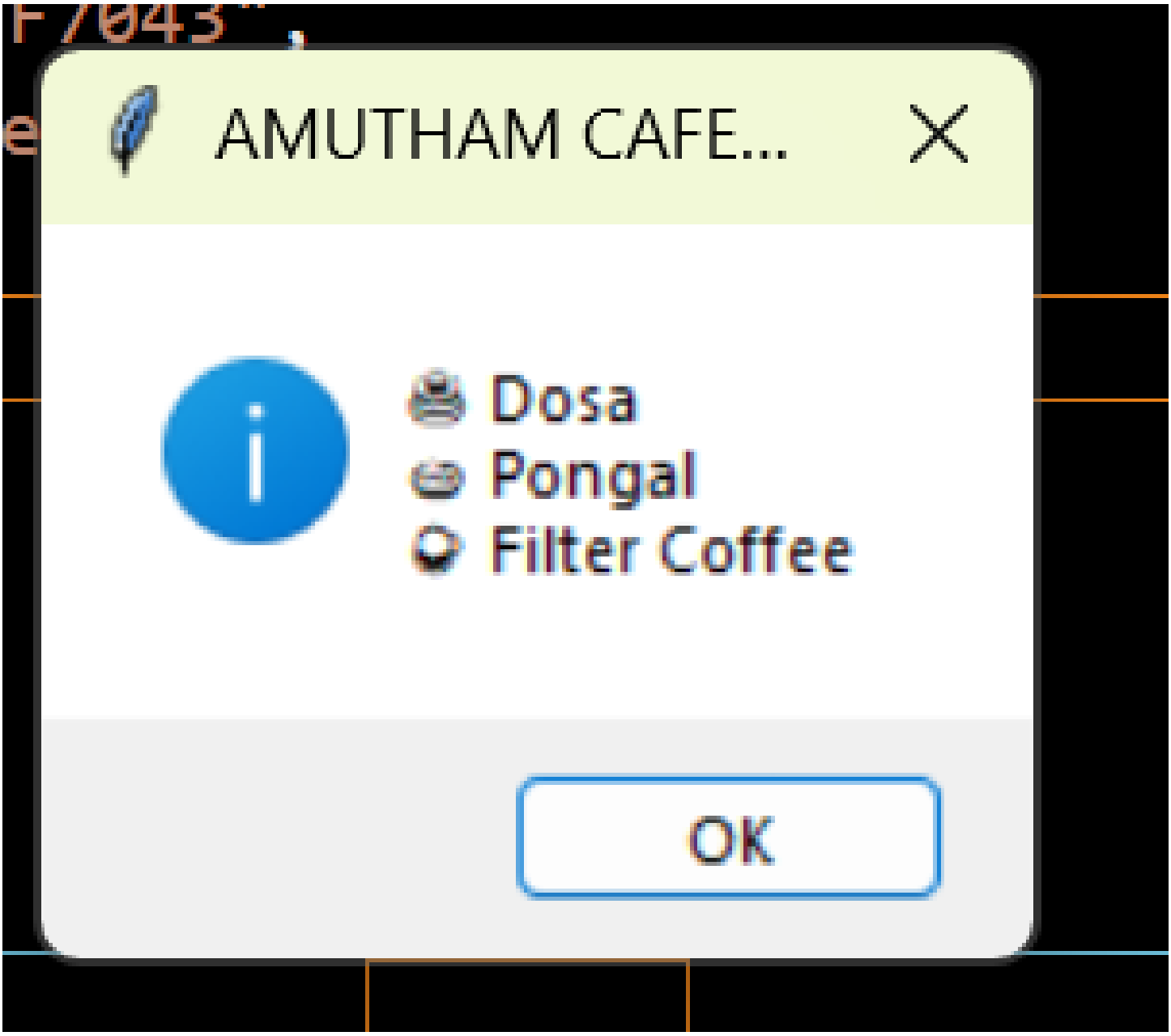
```
    choice = meal_var.get()
    if choice == "Breakfast":
        items = "Dosa\nPongal\nFilter Coffee"
    elif choice == "Lunch":
        items = "Meals Combo\nSambar Rice\nButtermilk"
    elif choice == "Snacks":
        items = "Samosa\nCake\nFresh Juice"
    else:
        items = "Please select a meal type."
    messagebox.showinfo("AMUTHAM CAFE Menu", items)
root = tk.Tk()
root.title("AMUTHAM CAFE")
root.geometry("350x250")
tk.Label(root, text="Welcome to AMUTHAM CAFE").pack(pady=10)
tk.Label(root, text="Select Meal Type:").pack()
meal_var = tk.StringVar()
tk.Radiobutton(root, text="Breakfast", variable=meal_var, value="Breakfast").pack()
tk.Radiobutton(root, text="Lunch", variable=meal_var, value="Lunch").pack()
tk.Radiobutton(root, text="Snacks", variable=meal_var, value="Snacks").pack()
tk.Button(root, text="Show Menu", command=show_menu).pack(pady=10)
root.mainloop()
```

How It Works

1. The program uses the Tkinter library to create a window interface.
2. Labels are used to display text in the window.
3. Radio buttons allow the user to select meal type.
4. When the button is clicked, the function show_menu() is executed.
5. Based on the selected option, menu items are displayed.
6. The output is shown using a message box popup.

OUTPUT:


AMUTHAM CAFE
Welcome to AMUTHAM CAFE
Select Meal Type:
Breakfast
Lunch
Snacks
Show Menu
46



VUI (Voice User Interface)

A Voice User Interface (VUI) allows users to interact with the system using voice commands. It provides hands-free and natural communication using speech recognition.

IMPLEMENTATION CODE:

```
import speech_recognition as sr

def show_menu(command):
    command = command.lower()
    print("\n--- AMUTHAM CAFE Voice Menu ---")
```

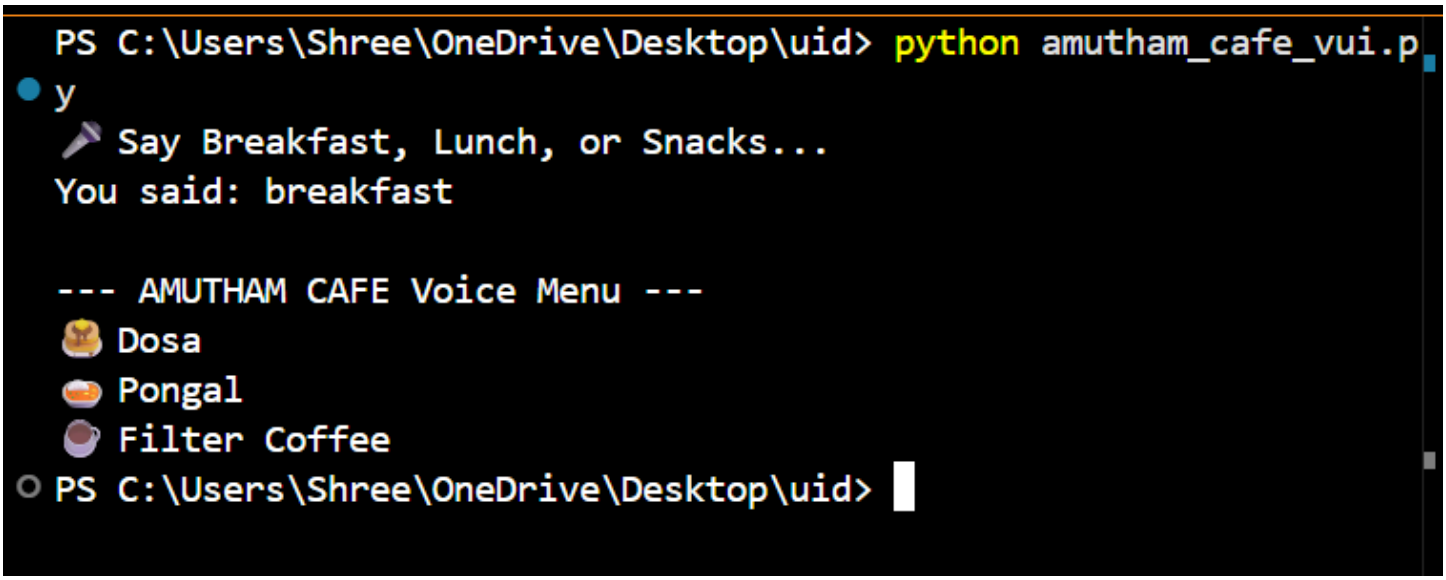
```
    if "breakfast" in command:
        print("Dosa")
        print("Pongal")
        print("Filter Coffee")
    elif "lunch" in command:
        print("Meals Combo")
        print("Sambar Rice")
        print("Buttermilk")
    elif "snacks" in command:
        print("Samosa")
        print("Cake")
        print("Fresh Juice")
    else:
        print("Please say Breakfast, Lunch, or Snacks.")
def listen():
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("Say Breakfast, Lunch, or Snacks...")
        recognizer.adjust_for_ambient_noise(source)
        audio = recognizer.listen(source)
    try:
        command = recognizer.recognize_google(audio)
        print("You said:", command)
        show_menu(command)
    except:
        print("Error in voice recognition.")
```

```
if __name__ == "__main__":

    listen()
```

How It Works:

1. The program uses the SpeechRecognition library.
2. The system listens to voice input through the microphone.
3. The voice input is converted into text using Google's speech recognition.
4. The recognized text is checked for keywords (Breakfast, Lunch, Snacks).
5. Based on the spoken command, menu items are displayed.
6. If speech is unclear, an error message is shown.

OUTPUT:



Conclusion

CLI is fast and efficient for technical users.
GUI is user-friendly and visually interactive.
VUI provides natural and hands-free interaction but depends on voice accuracy.