## LAB EXPERIMENT-3

## NAME:SHALINI R K ROLL NO:230701304

1.COMMAND LINE INTERFACE:

PYTHON CODE:

import os

import sys

def rename\_file(old\_name, new\_name):

try:

os.rename(old\_name, new\_name)

print(f"File renamed from {old\_name} to {new\_name}")

except FileNotFoundError:

print(f"Error: {old\_name} not found.")

except Exception as e:

print(f"An error occurred: {e}")

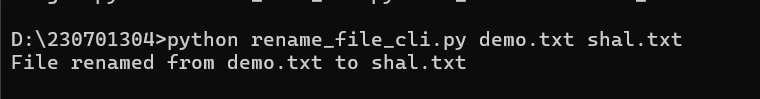
if \_\_name\_\_ == "\_\_main\_\_":

if len(sys.argv) != 3:

print("Usage: python rename\_file\_cli.py <old\_filename> <new\_filename>")

else:

rename\_file(sys.argv[1], sys.argv[2])  
OUTPUT:



2.GRAPHICAL USER INTERFACE:

PYTHON CODE:

import tkinter as tk

from tkinter import messagebox

import os

# Function to rename the file

def rename\_file():

old\_name = old\_filename\_entry.get().strip()

new\_name = new\_filename\_entry.get().strip()

# Check if input fields are empty

if not old\_name or not new\_name:

messagebox.showwarning("Warning", "Please enter both filenames!")

return

# Check if the old file exists

if not os.path.exists(old\_name):

messagebox.showerror("Error", f"File '{old\_name}' not found.")

return

# Check if the new file already exists

if os.path.exists(new\_name):

overwrite = messagebox.askyesno("Warning", f"'{new\_name}' already exists. Overwrite?")

if not overwrite:

return

try:

os.rename(old\_name, new\_name)

messagebox.showinfo("Success", f"File renamed from '{old\_name}' to '{new\_name}'")

except Exception as e:

messagebox.showerror("Error", f"An error occurred: {e}")

# Create main window

root = tk.Tk()

root.title("File Renamer")

root.geometry("400x150")

root.resizable(False, False) # Fixed window size

# Labels

tk.Label(root, text="🗂️ Old Filename:", font=("Arial", 10)).grid(row=0, column=0, padx=10, pady=5, sticky="w")

tk.Label(root, text="📝 New Filename:", font=("Arial", 10)).grid(row=1, column=0, padx=10, pady=5, sticky="w")

# Entry fields

old\_filename\_entry = tk.Entry(root, width=40)

old\_filename\_entry.grid(row=0, column=1, padx=10, pady=5)

new\_filename\_entry = tk.Entry(root, width=40)

new\_filename\_entry.grid(row=1, column=1, padx=10, pady=5)

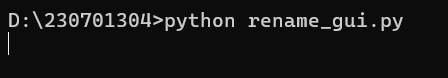
# Styled Button

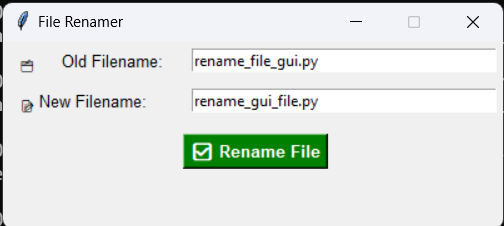
rename\_button = tk.Button(root, text="✅ Rename File", bg="green", fg="white", font=("Arial", 10, "bold"), command=rename\_file)

rename\_button.grid(row=2, column=0, columnspan=2, pady=10)

# Run the GUI event loop

root.mainloop()  
OUTPUT:





3.VOICE USER INTERFACE:

PYTHON CODE:

import speech\_recognition as sr

import os

def rename\_file\_from\_voice\_command(command):

try:

words = command.lower().split(" ")

if "rename" in words and "to" in words:

rename\_index = words.index("rename")

to\_index = words.index("to")

# Extract old and new filenames

old\_name = words[rename\_index + 1]

new\_name = words[to\_index + 1]

# Check if file exists

if not os.path.exists(old\_name):

print(f"Error: File '{old\_name}' not found.")

return

# Rename file

os.rename(old\_name, new\_name)

print(f"✅ File renamed from '{old\_name}' to '{new\_name}'")

else:

print("Invalid command format. Say: 'Rename oldfile.txt to newfile.txt'")

except Exception as e:

print(f"Error: {e}")

def listen\_for\_command():

recognizer = sr.Recognizer()

mic = sr.Microphone()

print("🎤 Listening for command to rename a file...")

with mic as source:

recognizer.adjust\_for\_ambient\_noise(source)

audio = recognizer.listen(source)

try:

command = recognizer.recognize\_google(audio)

print(f"🎙️ Command received: {command}")

rename\_file\_from\_voice\_command(command)

except sr.UnknownValueError:

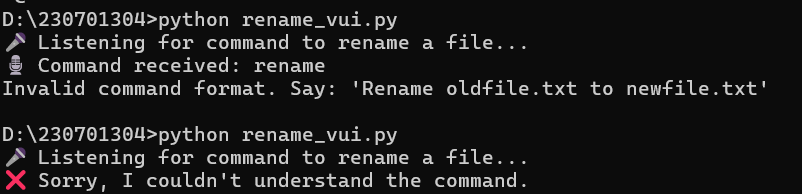
print("❌ Sorry, I couldn't understand the command.")

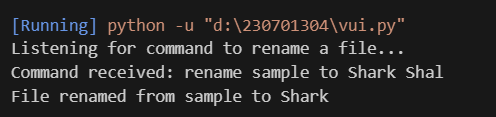
except sr.RequestError as e:

print(f"⚠️ Could not request results from Google Speech Recognition service; {e}")

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

listen\_for\_command()





4.USER SATISFACTION COMPARISON:

PYTHON CODE:

def survey():

print("Rate your satisfaction with the following interfaces (1-5):")

# Get user input for each interface

try:

cli\_satisfaction = int(input("CLI (1-5): "))

gui\_satisfaction = int(input("GUI (1-5): "))

vui\_satisfaction = int(input("VUI (1-5): "))

# Ensure valid ratings

if not (1 <= cli\_satisfaction <= 5 and 1 <= gui\_satisfaction <= 5 and 1 <= vui\_satisfaction <= 5):

print("Please enter ratings between 1 and 5 only.")

return

# Display the ratings

print("\nYour satisfaction ratings:")

print(f"CLI: {cli\_satisfaction}")

print(f"GUI: {gui\_satisfaction}")

print(f"VUI: {vui\_satisfaction}")

# Calculate the average satisfaction

avg\_satisfaction = (cli\_satisfaction + gui\_satisfaction + vui\_satisfaction) / 3

print(f"\nAverage Satisfaction Score: {avg\_satisfaction:.2f}")

except ValueError:

print("Invalid input! Please enter numbers between 1 and 5.")

# Run the survey function

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

survey()

