UID EXPERIMENT 3

NAME: Durkasri S M

ROLL NO: 240701132

1. COMMAND LINE INTERFACE (CLI)

```
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: File '{old_name}' not found.")
    except PermissionError:
        print("Error: Permission denied.")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

def main():
    if len(sys.argv) != 3:
        print("Usage: python rename_file_cli.py <old_filename> <new_filename>")
        sys.exit(1)

    old_name = sys.argv[1]
    new_name = sys.argv[2]

    rename_file(old_name, new_name)
```

```
if __name__ == "__main__":
    main()
```

OUTPUT:

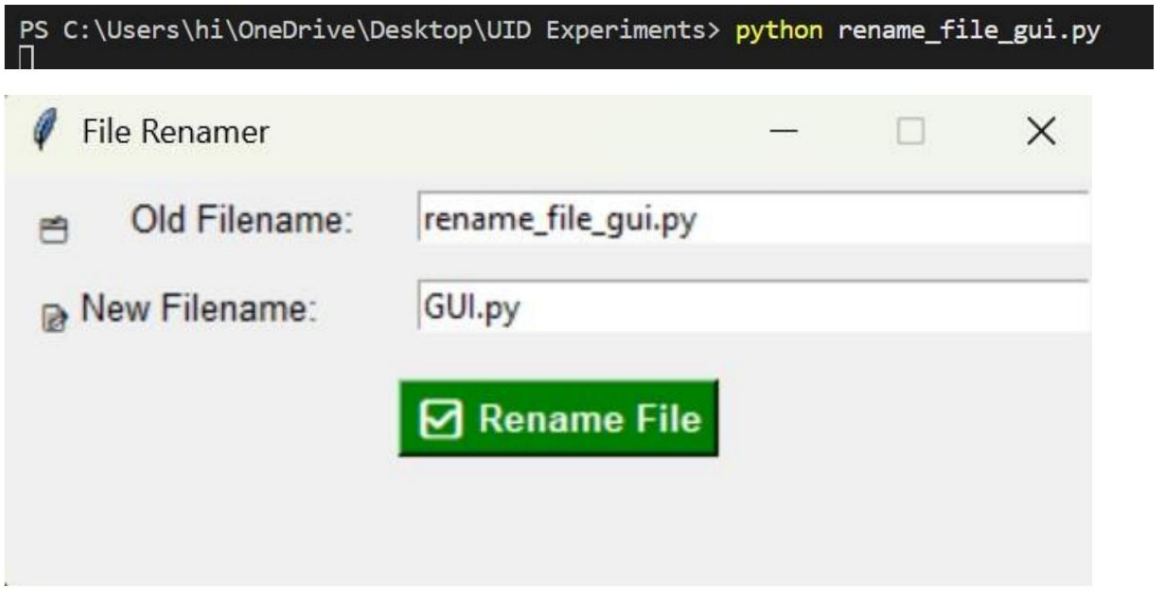
2. GRAPHICAL USER INTERFACE (GUI)

```
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)

# 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:

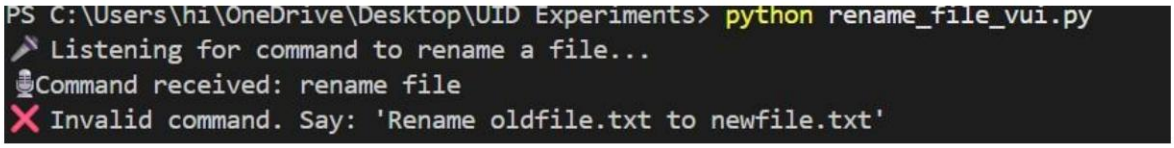


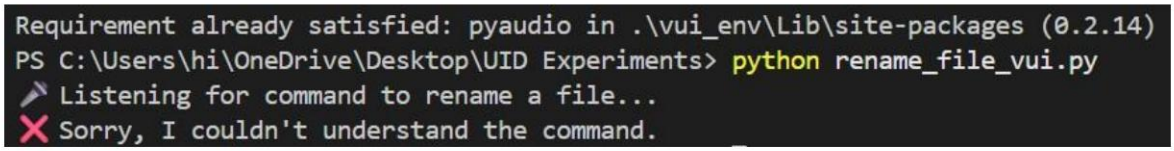
3. VOICE USER INTERFACE (VUI)

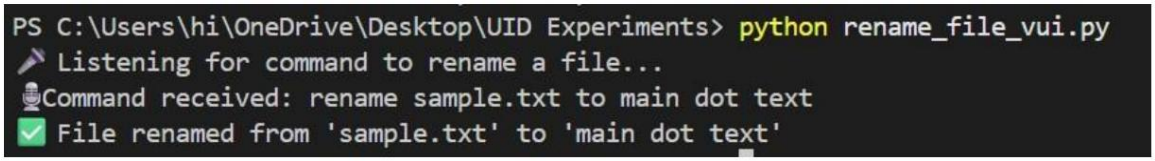
```
import speech_recognition as sr
import os

def rename_file_from_voice_command(command):
  try:
    command = command.lower()

    if "rename" not in command or "to" not in command:
      print("Invalid command. Say: 'Rename oldfile.txt to newfile.txt'")
      return

    # Extract filenames
    before_to, after_to = command.split("to", 1)
    old_name = before_to.replace("rename", "").strip()
    new_name = after_to.strip()

    # Check if old file exists
    if not os.path.exists(old_name):
      print(f"Error: File '{old_name}' not found.")
      return

    # Check if new file already exists
    if os.path.exists(new_name):
      print(f"Error: '{new_name}' already exists.")
      return

    os.rename(old_name, new_name)
```

```
            print(f"File renamed from '{old_name}' to '{new_name}'")

        except Exception as e:
            print(f"Error: {e}")

def listen_for_command():
    recognizer = sr.Recognizer()

    try:
        with sr.Microphone() as source:
            print("Listening for command to rename a file...")
            recognizer.adjust_for_ambient_noise(source)
            audio = recognizer.listen(source)

        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"Speech recognition service error: {e}")
    except OSError:
        print("Microphone not found or access denied.")

if __name__ == "__main__":
    listen_for_command()
```

OUTPUT:


PS C:\Users\hi\OneDrive\Desktop\UID Experiments> python rename_file_vui.py
🎤 Listening for command to rename a file...
🎙️Command received: rename file
❌ Invalid command. Say: 'Rename oldfile.txt to newfile.txt'


Requirement already satisfied: pyaudio in .\vui_env\Lib\site-packages (0.2.14)
PS C:\Users\hi\OneDrive\Desktop\UID Experiments> python rename_file_vui.py
🎤 Listening for command to rename a file...
❌ Sorry, I couldn't understand the command.


PS C:\Users\hi\OneDrive\Desktop\UID Experiments> python rename_file_vui.py
🎤 Listening for command to rename a file...
🎙️Command received: rename sample.txt to main dot text
✅ File renamed from 'sample.txt' to 'main dot text'

4. USER SATISFACTION COMPARISON

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

  try:
    cli_satisfaction = int(input("CLI (1-5): "))
    gui_satisfaction = int(input("GUI (1-5): "))
    vui_satisfaction = int(input("VUI (1-5): "))

    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

    print("\nYour satisfaction ratings:")
    print(f"CLI: {cli_satisfaction}")
    print(f"GUI: {gui_satisfaction}")
    print(f"VUI: {vui_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.")


if __name__ == "__main__":
    survey()
```

OUTPUT: