EXP NO:3

**CLI (Command Line Interface)**

For the **CLI**, you can provide a Python script that renames a file via command-line arguments. Here's how you could implement it:

**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}")

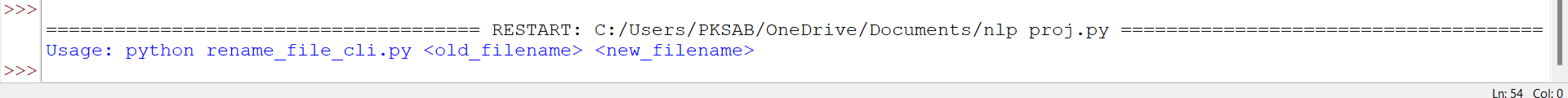
if \_\_name\_\_ == "\_\_main\_\_":

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])



**GUI (Graphical User Interface)**

In the **GUI**, we will use **Tkinter** to create a simple interface where the user can type the old and new filenames.

import tkinter as tk

from tkinter import messagebox

import os

def rename\_file():

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

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

try:

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

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

except FileNotFoundError:

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

except Exception as e:

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

# Set up the main window

root = tk.Tk()

root.title("File Renamer")

# Create and place labels, entries, and buttons

tk.Label(root, text="Old Filename:").grid(row=0, column=0)

tk.Label(root, text="New Filename:").grid(row=1, column=0)

old\_filename\_entry = tk.Entry(root)

old\_filename\_entry.grid(row=0, column=1)

new\_filename\_entry = tk.Entry(root)

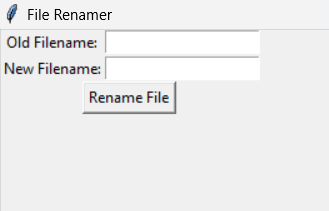
new\_filename\_entry.grid(row=1, column=1)

rename\_button = tk.Button(root, text="Rename File", command=rename\_file)

rename\_button.grid(row=2, columnspan=2)

# Start the Tkinter event loop

root.mainloop()



**VUI (Voice User Interface)**

For the **VUI**, we can use the **SpeechRecognition** library to listen for voice commands. The user will say something like, "Rename oldfilename.txt to newfilename.txt."

import speech\_recognition as sr

import os

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

# Extracting old and new filename from the command

try:

words = command.split(" ")

old\_name = words[1]

new\_name = words[3]

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()

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()