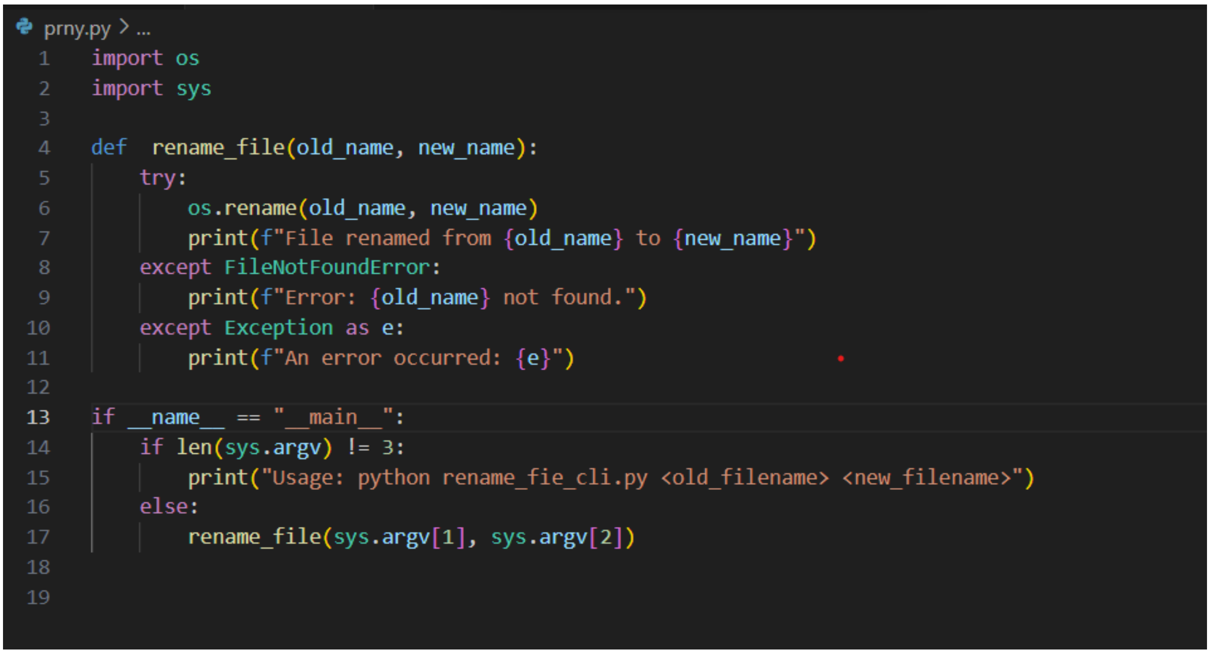
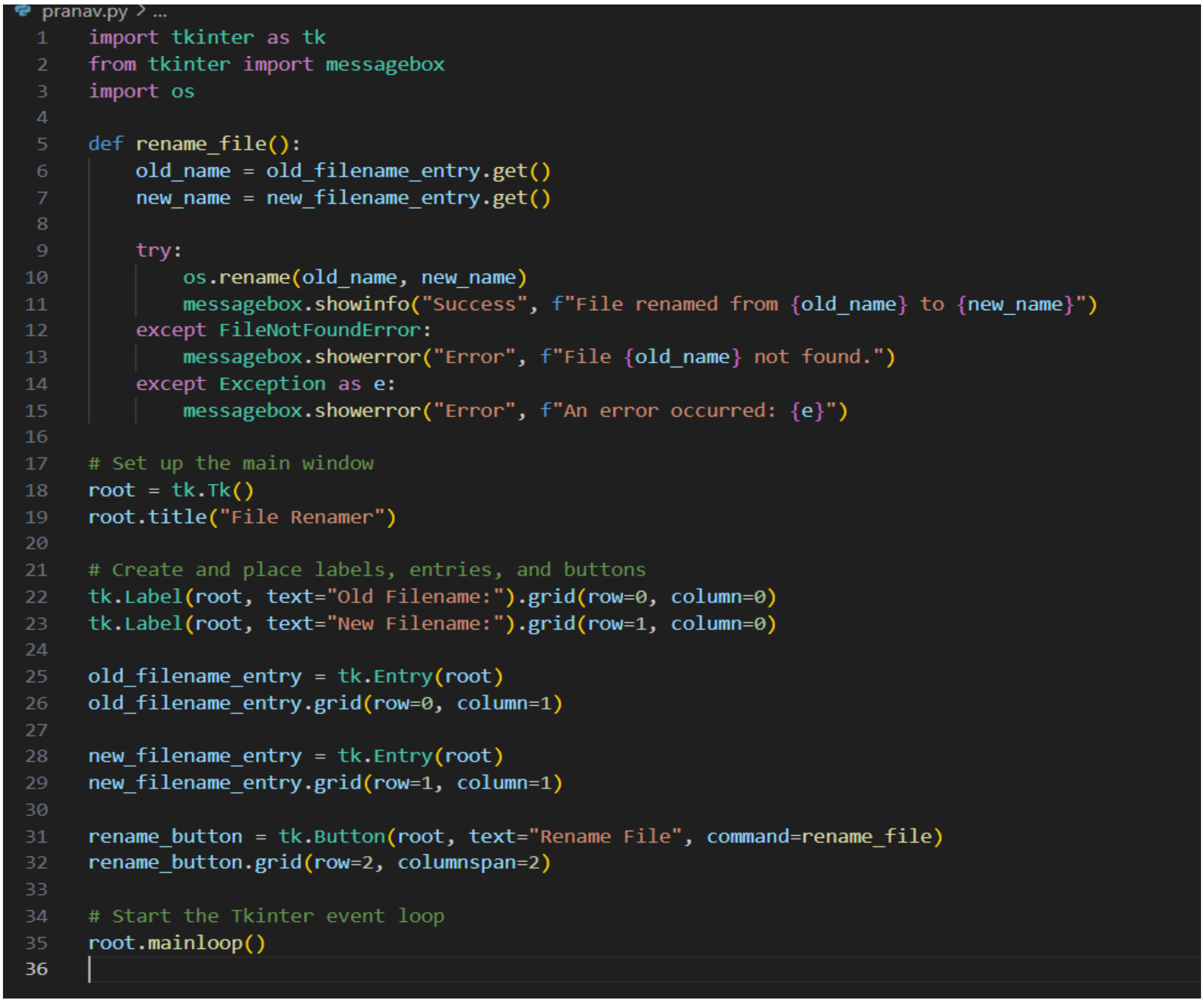
1.CLI(Command Line Interface)



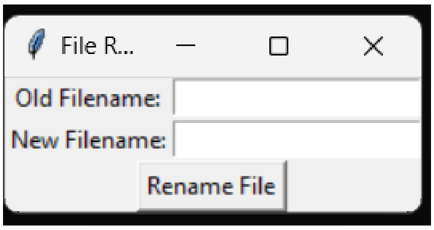
```python
import os
import sys

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_fie_cli.py <old_filename> <new_filename>")
    else:
        rename_file(sys.argv[1], sys.argv[2])


```

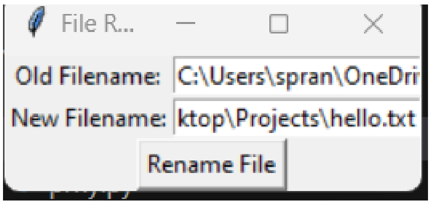
2.GUI(Graphical User Interface)



```python
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()
```

Output:

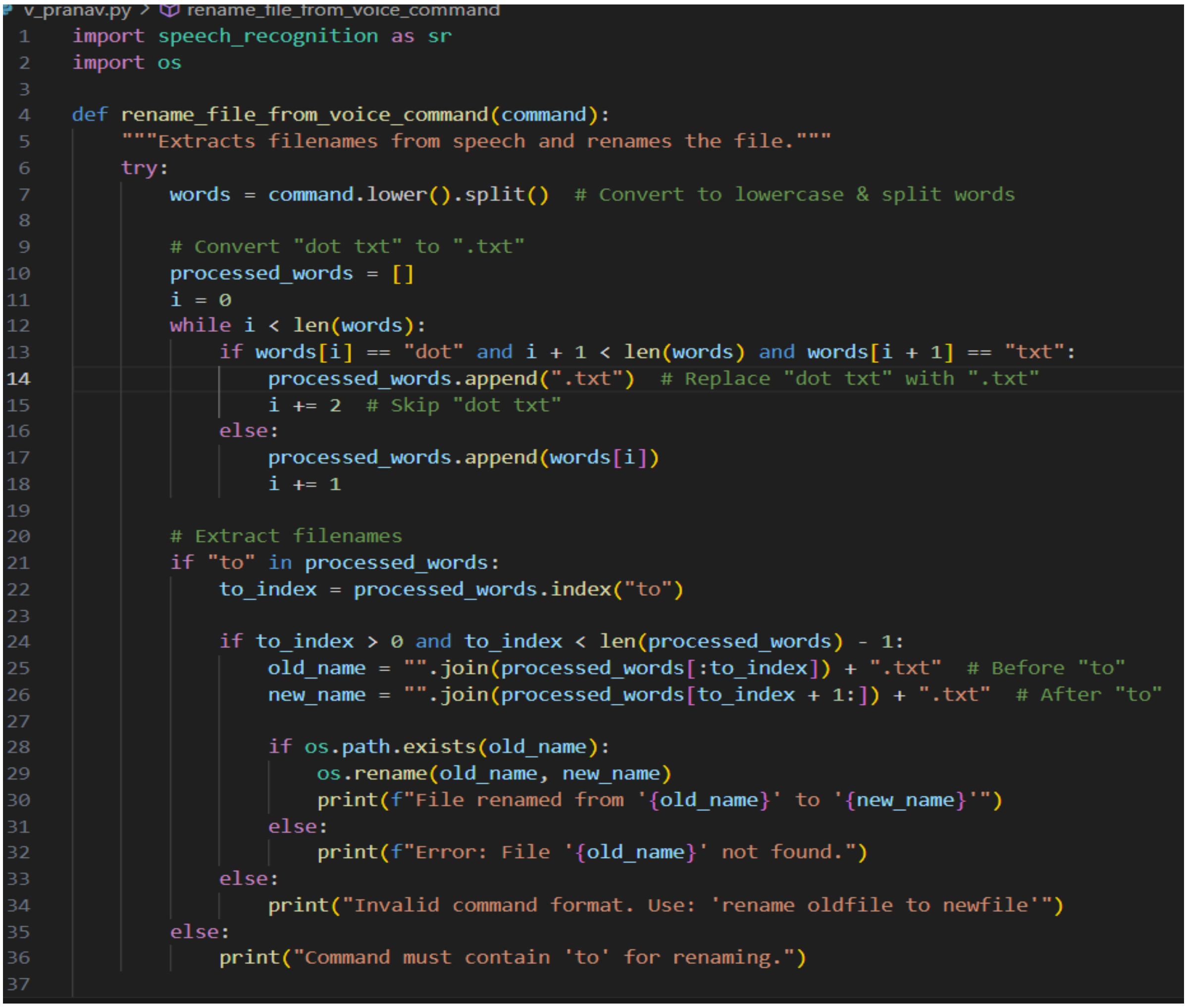




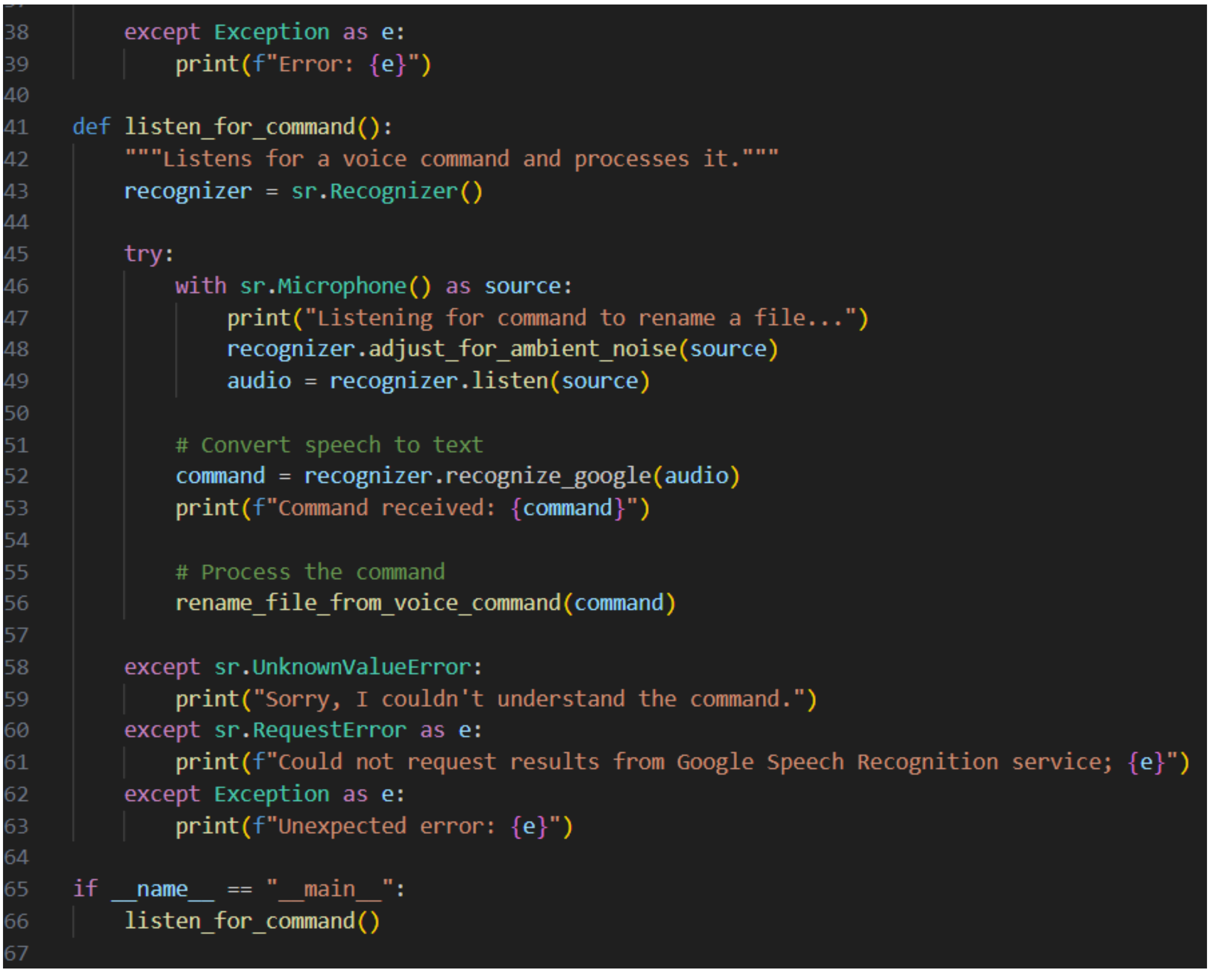
Old Filename: C:\Users\spran\OneDri

New Filename: ktop\Projects\hello.txt

3.VUI(Voice User Interface)



```python
import speech_recognition as sr
import os

def rename_file_from_voice_command(command):
    """Extracts filenames from speech and renames the file."""
    try:
        words = command.lower().split()  # Convert to lowercase & split words

        # Convert "dot txt" to ".txt"
        processed_words = []
        i = 0
        while i < len(words):
            if words[i] == "dot" and i + 1 < len(words) and words[i + 1] == "txt":
                processed_words.append(".txt")  # Replace "dot txt" with ".txt"
                i += 2  # Skip "dot txt"
            else:
                processed_words.append(words[i])
                i += 1

        # Extract filenames
        if "to" in processed_words:
            to_index = processed_words.index("to")

            if to_index > 0 and to_index < len(processed_words) - 1:
                old_name = "".join(processed_words[:to_index]) + ".txt"  # Before "to"
                new_name = "".join(processed_words[to_index + 1:]) + ".txt"  # After "to"

                if os.path.exists(old_name):
                    os.rename(old_name, new_name)
                    print(f"File renamed from '{old_name}' to '{new_name}'")
                else:
                    print(f"Error: File '{old_name}' not found.")
            else:
                print("Invalid command format. Use: 'rename oldfile to newfile'")
        else:
            print("Command must contain 'to' for renaming.")

```



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

def listen_for_command():
    """Listens for a voice command and processes it."""
    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)

        # Convert speech to text
        command = recognizer.recognize_google(audio)
        print(f"Command received: {command}")

        # Process the 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}")
    except Exception as e:
        print(f"Unexpected error: {e}")

if __name__ == "__main__":
    listen_for_command()

```