USER INTERFACE DESIGN EXPERIMENT-3

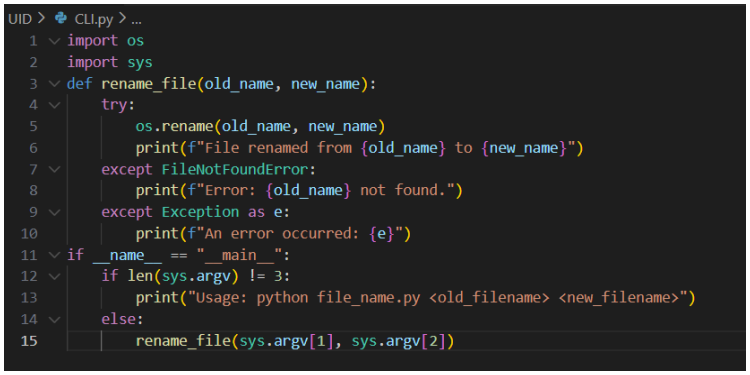
Difference between CLI(Command line interface) and GUI(Graphical user interface) and VUI(voice user interface)

Command line Interface(CLI)

- A Text-based Interface (Command line interface) enables users to communicate with a computer using the command line, as opposed to using a graphical interface.
- Compact and Powerful – CLI is usually faster and more powerful than a GUI, particularly for expert users who know the correct commands.
- Automation & Scripting – The command-line interface is there to allow users to automate tasks with scripts, making the repetitive tasks easier.
- Lightweight and Resource Friendly – Compared to GUI application, CLI uses very little system resources which is why they are most suited for use on remote servers

example: Windows command prompt.

IMPLEMENTATION



Output

```
(.venv) PS C:\Users\Sivaguru\OneDrive\Desktop\Sg\UID> python CLI.py hi.txt ji.txt
Error: hi.txt not found.
(.venv) PS C:\Users\Sivaguru\OneDrive\Desktop\Sg\UID> python CLI.py hi.html ji.html
File renamed from hi.html to ji.html
```

Graphical user interface(GUI)

- A GUI uses visual elements including icons, buttons, and windows which makes it more user-friendly for users to work with the system.
- More Resource Intensive – GUI applications use more system resources (CPU, RAM, and GPU) in contrast to a Command Line Interface (CLI) that can degrade performance in low-end devices.
- Limited Automation: GUI does not have automation and scripting abilities like the CLI and is thus more tedious for repetitive jobs.

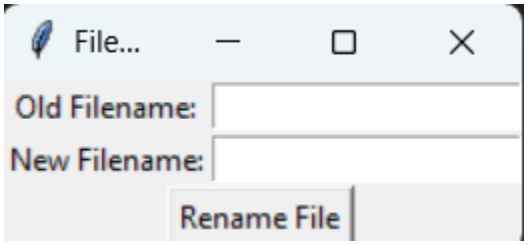
Example: Windows, Android, iOS

IMPLEMENTATION

UID > GUI.py > ...

```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"Error: {old_name} not found.")
    except Exception as e:
        messagebox.showerror("Error", f"An error occurred: {e}")
root = tk.Tk()
root.title("File Renamer")
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)
root.mainloop()
```

Output



Voice user interface(VUI)

- No Hands – VUI allows me to talk to a system and in some cases this is much more convenient than having to type. • Natural Communication– VUI allows for more natural and fluid communication, as it can process users' speech and verbal commands, eliminating typing or clicking.
- Accuracy Challenges – Speech recognition can occasionally have difficulty with accents, background noise, and complex commands, sometimes resulting in misinterpretation.

Example: Amazon, Alexa, Google Assistant

IMPLEMENTATION

UID > vui1.py > listen_for_filename

```python
import speech_recognition as sr
import os
def rename_file_from_voice_command(old_name, new_name):
    try:
        old_name = old_name + ".txt"
        new_name = new_name + ".txt"
        if not os.path.exists(old_name):
            print(f"X Error: {old_name} not found.")
            return
        os.rename(old_name, new_name)
        print(f" File successfully renamed from {old_name} to {new_name}")
    except Exception as e:
        print(f" Error: {e}")
def listen_for_filename (prompt):
    """Listens for a single filename input via voice command."""
    recognizer = sr.Recognizer()
    mic = sr.Microphone()
    with mic as source:
        recognizer.adjust_for_ambient_noise(source, duration=3) # Increase noise adaptation
        print(f" {prompt}")
        try:
            audio = recognizer.listen(source, timeout=10, phrase_time_limit=5) # Increased timeout
            command = recognizer.recognize_google(audio, language="en-US")
            print(f" You said: {command}")
            return command.strip().replace(" ", "_") # Replace spaces with underscores
        except sr.UnknownValueError:
            print("X Could not understand. Please try again.")
            return None
        except sr.WaitTimeoutError:
            print(" Timeout: No speech detected. Try speaking louder and clearly.")
            return None
```

```python
if __name__ == "__main__":
    print(" Welcome to the Voice-Controlled File Renamer!")
    old_name = None
    while old_name is None:
        old_name = listen_for_filename("Say the name of the file you want to rename (without .txt)")
        new_name = None
    while new_name is None:
        new_name = listen_for_filename("Say the new name for the file (without.txt)")
        rename_file_from_voice_command(old_name, new_name)
```

Output

```
(.venv) PS C:\Users\Sivaguru\OneDrive\Desktop\Sg\UID> python vui1.py
 Welcome to the Voice-Controlled File Renamer!
 Say the name of the file you want to rename (without .txt)
 You said: sample
 Say the new name for the file (without.txt)
 You said: hello
 File successfully renamed from sample.txt to hello.txt
```