ASSISTANT_NILU – Technical Documentation

Version: 1.0

Author: NISARG CHAUDHARY AKA NILUOP

Repo: https://github.com/NILUOP/ASSISTANT_NILU

Table of Contents

Overview

ASSISTANT_NILU is a voice-controlled desktop assistant built using Python and OpenAIs Whisper model. It supports natural command execution, music playback, weather updates, web automation, and system-level actions.

Features

- Hotword Recognition: Starts only when the user says hello, hi, or nilu

- Voice-to-Text: Uses Whisper AI for highly accurate transcription

- Text-to-Speech: Feedback provided using pyttsx3

- Music Playback: Local music with shuffle, next/prev, pause/resume support

- Weather Forecast: Current and 3-day GUI forecast via WeatherAPI

- Screenshot: Takes desktop screenshots on command

- Volume Control: Increase/decrease/mute/unmute via voice

- Web Integration: Opens Google, YouTube, LinkedIn, GitHub, ChatGPT, and more

- Modular Design: Each function is decoupled and scalable

Project Structure

```
ASSISTANT_NILU/
├── main.py
├── recorder.py
├── transcriber.py
├── speaker.py
├── command_processor.py
├── music_plyer.py
├── music_list.py
├── weather_updater.py
├── volume_changer.py
├── requirements.txt
├── README.md
```

Setup and Installation

Prerequisites

- Python 3.8+
- CUDA-enabled GPU (optional for Whisper)
- FFmpeg installed

Install Dependencies

```
git clone https://github.com/NILUOP/ASSISTANT_NILU.git
cd ASSISTANT_NILU
pip install -r requirements.txt
```

Configure API Keys

Create a file api.py:

```
weather_api = "YOUR_WEATHER_API_KEY"
```

How It Works

1. Assistant listens for hotwords
2. Prompts Ready to take command
3. Records voice for command
4. Uses Whisper to transcribe
5. Routes command and executes it
6. Responds with voice

Voice Flow Example

You: Hi Nilu

Assistant: Ready to take command

You: Play music

Assistant: Plays and says Playing: [Song Name]

Modules Description

Module	Description
main.py	Main entry point and loop
recorder.py	Records audio clips
transcriber.py	Converts audio to text via Whisper
speaker.py	Text-to-speech using pyttsx3
command_processor.py	Detects and routes command

music_plyer.py	Plays music using VLC
music_list.py	Contains song file paths
weather_updater.py	Weather data fetch + forecast GUI
volume_changer.py	Controls system audio levels

Commands Reference

Command Example	-	Action
"Play music"	-	Start playback
"Pause the song"	-	Pause music
"Play [song name]"	-	Play song by name
"Next song" / "Previous song"	-	Skip/rewind
"Shuffle on/off"	-	Toggle shuffle
"Open YouTube/Google/..."	-	Open in browser
"Search for [term] on YouTube/Google"	-	Google/YouTube search
"Take screenshot"	-	Capture screen
"Increase volume by 50"	-	Raise volume
"Mute"/"Unmute"	-	Toggle mute
"Weather in [City]"	-	Speak weather
"Forecast for [City]"	-	Show GUI forecast

Dependencies

- openai-whisper
- pyttsx3
- sounddevice
- scipy

- requests

- tkinter

- vlc

- pyautogui

- pycaw

- plyer

- gTTS

Future Improvements

- Wake word detection

- Full GPT conversation

- Calendar/reminders

- Cross-platform support

- Multilingual

- GUI dashboard

Contact

Open an issue or connect via LinkedIn for ideas or collaboration.