# Voice-basedNotesandMemo Systems

## Introduction:

In today's fast-paced digital age, traditional methods of note-taking and memo recording are being rapidly replaced by smart, efficient, and accessible technologies. Voice-based systems have emerged as a powerful solution to enhance productivity and convenience. The "Voice-based Notes and Memo System" aims to provide users with a seamless method to record spoken thoughts, ideas, and reminders, converting them into textual notes automatically using speech recognition technologies.

This project is especially beneficial for professionals, students, and individuals whoprefer hands-free interaction with devices or need to capture ideas quickly on the go.With advancements in voice processing, it's now feasible to transcribe speech with high accuracy and store it systematically for future reference.

## ProblemStatement:

Manual note-taking using traditional input devices like keyboards or writing tools can be time-consuming, error-prone, and inefficient in situations where usersare multitasking or physically constrained. Moreover, people often forget ideas if they cannot record them immediately. There's a clear need for a reliable and user-friendly system that can:

* Recordvoiceinputs.
* Convertspeechtotext.
* Savethe convertedtextinanorganizedmanner(e.g.,CSVortextfile).
* Allowappendingnewentrieswithoutoverwritingpreviousones.

The "Voice-based Notes and Memo System" addresses this issue by providing a lightweight, intuitive, and accurate voice-to-text platform that updates the user's notes automatically.

## Objectives

Thekeyobjectivesofthisprojectinclude:

* Todevelopasystemthatallowsuserstoinputmemosornotesusing voice.
* Toconvert`.wav` audiofilesintotranscribedtextusingspeech recognition libraries.
* Tostore thetranscriptionsinastructuredformatlike`.csv`or`.txt`without recreating the file each time.
* TosupportbatchprocessingofmultipleaudiofilesuploadedviaaZIParchive.
* TomakethesystemusableinenvironmentslikeGoogleColabforcloud-based accessibility.

## Methodology

The development of the Voice-based Notes and Memo System follows a structured methodology to ensure accurate voice recognition, efficient storage, and seamless user experience. The steps are as follows:

### Audio Input Collection

The user records their note or memo as a .wav audio file using any voice recorder. This allows flexibility in capturing voice inputs even without a live microphone.

### AudioFileUpload

Thesystem allows users to upload one or more .wavfiles at atime. These files are collected through a file input interface and stored temporarily for processing.

### SpeechRecognition

Using a speech recognition library (such as Python’s speech\_recognitionmodule), the system processes each .wav file to extract the spoken words and convert them into text format. This module uses Google's Web Speech API for accurate transcription.

### DataValidationandCleaning

The transcribed text is cleaned to remove background noise errors, common artifacts, or misinterpretations. Optional keyword filtering can be implemented to ignore irrelevant inputs.

### FileHandlingandStorage

* + Iftheoutputfiledoesnot exist, itiscreated.
  + Ifit alreadyexists,thesystemappendsthenewnotewithoutoverwriting the previous entries.
  + Each recognizednoteissavedinanewlineorrow,maintainingthe chronological order of in put.

### ErrorHandling

Thesystemincludesexceptionhandlingtomanagecasessuchas:

* + Unrecognizableaudio
  + Missingfiles
  + Fileformatissues

### OutputGeneration

The recognized text is saved and can be reviewed later as a memo or note. The outputiseithera .txtfile for simple notesora.csvfile if metadata (like date and time) needs to be recorded alongside the note.

### RepeatableUse

The system is designed to be used multiple times. Each run will append new content to the existing file without duplication, ensuring continuity.\

## ProgramFlow

!pipinstallSpeechRecognitionpandas

!pipinstallpandaspyttsx3

!apt-getinstall-yportaudio19-dev

!pipinstallpyaudiopip install pipwin

fromgoogle.colabimportfiles

print("Upload yourZIPfilecontaining.wavfiles:") uploaded = files.upload()

import osimportzipfile

extract\_folder = "extracted\_audio" os.makedirs(extract\_folder, exist\_ok=True)

forzip\_filenameinuploaded.keys():

ifzip\_filename.endswith('.zip'):

withzipfile.ZipFile(zip\_filename, 'r') aszip\_ref: zip\_ref.extractall(extract\_folder)

print(f"Extracted ZIPfileto folder: {extract\_folder}") else:

print(f"❌{zip\_filename}isnotaZIPfile.")

print("Files extracted:") print(os.listdir(extract\_folder)) import os

importzipfile

from google.colab import files importspeech\_recognitionassr

#Step1:Upload and unzip

uploaded=files.upload()#Uploadazipwith.wavfiles zip\_file\_name = list(uploaded.keys())[0]

extract\_path="Notes"

withzipfile.ZipFile(zip\_file\_name, 'r') aszip\_ref: zip\_ref.extractall(extract\_path)

print(f"\n✅Extractedto'{extract\_path}'")

#Step2:Findall.wavfilesinallsubfolderswav\_files = []

forroot,dirs, files\_in\_dir inos.walk(extract\_path): for file in files\_in\_dir:

if file.lower().endswith(".wav"): wav\_files.append(os.path.join(root, file))

ifnotwav\_files:

print("❌No.wavfilesfound.CheckyourZIPstructure.") else:

print("🎧Foundthefollowing .wavfiles:") for wf in wav\_files:

print("-",wf)

#Step3:Recognizeandwritetooutputfile recognizer = sr.Recognizer()

output\_file="Notes\_output.txt"

withopen(output\_file, "w",encoding="utf-8")asf: for wav\_path in wav\_files:

filename = os.path.basename(wav\_path) print(f"\n🎧Processing {filename}...") try:

with sr.AudioFile(wav\_path) as source: audio\_data=recognizer.record(source)

command=recognizer.recognize\_google(audio\_data) print(f"✅ Recognized: {command}") f.write(f"{filename}: {command}\n")

exceptsr.UnknownValueError:

print(f"⚠Couldnotunderstandaudioin{filename}.") f.write(f"{filename}:Couldnotunderstandaudio.\n")

exceptsr.RequestErrorase:

print(f"❌APIerrorfor{filename}:{e}") f.write(f"{filename}: API error - {e}\n")

print(f"\n🎧Allrecognizedcommandssavedto'{output\_file}'")

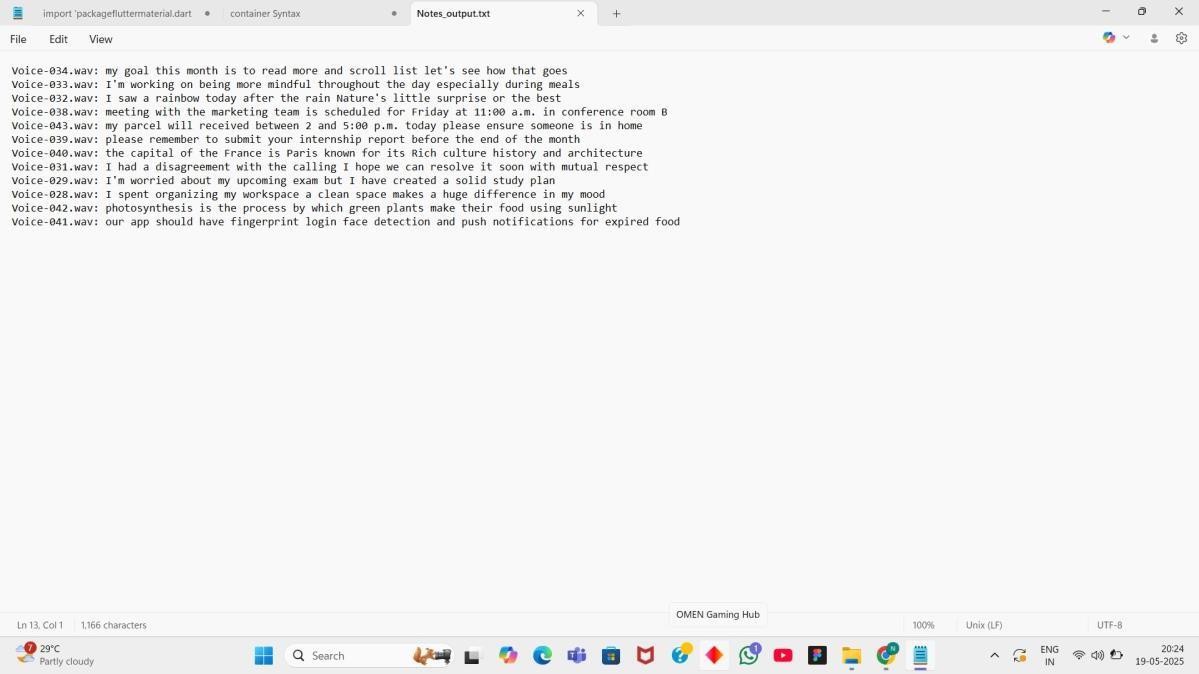
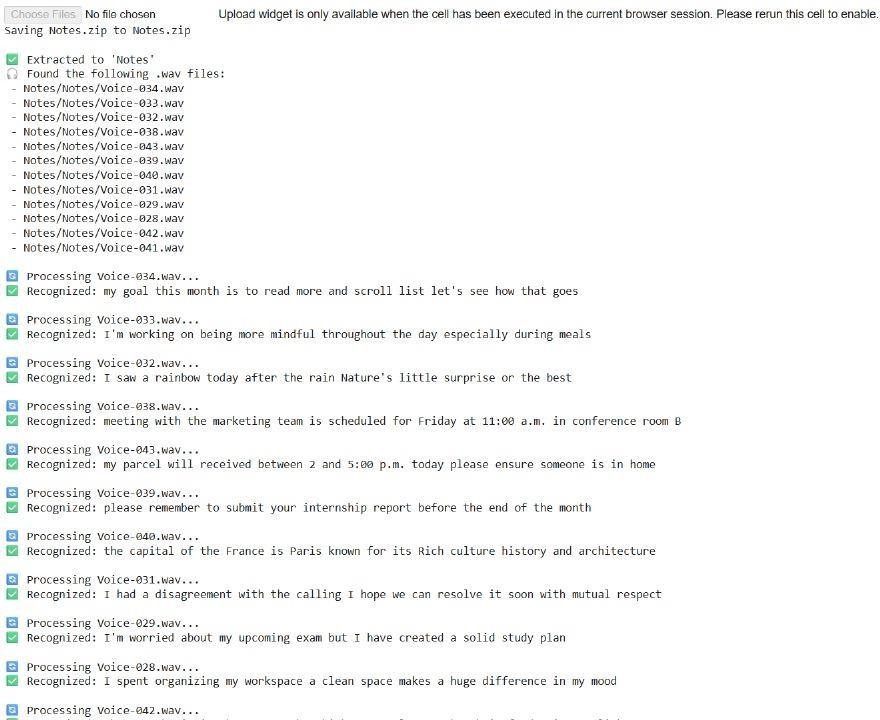
**InputFormat**

* `.wav`audiofiles(voicememosornotes)
* Combined ina`.zip`filefor batchupload

**OutputFormat**

* `.txt`file(or`.csv`ifneeded),containingeachtranscribednoteonanewline
* Automaticallyappended—nodataisoverwritten

## output:



**TestingandValidation:**

* + Testedwithvariedaccentsandsentence structures.
  + Includedbackgroundnoise insome filestotestrecognitionrobustness.
  + Validatedfileappendingfunctionalityandensurednooverwritingoccurred.
  + Ensuredsystemhandledunrecognizedaudio gracefullywithplaceholdertext.

## Conclusion:

The Voice-based Notes and Memo System successfully delivers an accessible, accurate, and convenient method for voice-based note-taking. By leveraging Python's speech recognition libraries and simple file handling mechanisms, the system can process multiple audio files, extract meaningful content, and store it systematically.

Thisapproachis highly extensible and can be adapted for mobile apps, accessibility tools, or productivity platforms. Future improvements may include live recordingsupport, multilingual support, speaker identification, and integration with cloud storage.