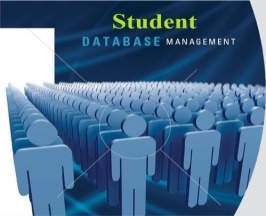
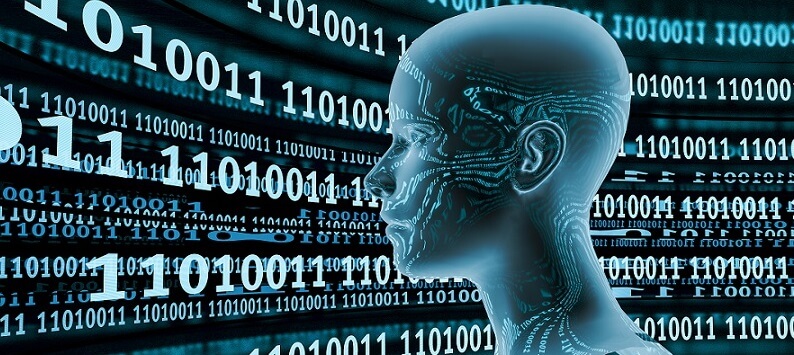
Maulana Abdul Kalam Azad University of Technology, (MAKAUT,WB)



**Project Report :**

VOICE ENABLED STUDENT DATABASE SYSTEM

BY

PAPPU KUMAR PASHI ( IT-3rd YEAR)

**In**

**Machine Learning Using Python**

Under the Guidance

Of SHUBHADEEP CHAKRABORTY



**DEPARTMENT OF IT**

**YEAR OF SUBMISSION - 2019**

MACHINE LEARNING USING PYTHON

****

**Supported by**

**Webskitters Academy**

Session : 2019-2020

An Industrial Training Report Submitted To

BENGAL COLLEGE OF ENGINEERING AND TECHNOLOGY

**Towards Partial fulfillment of Degree of**

**BachelorofInformationTechnology**

**In**

**Information Technology**

**Submitted To :-**

**Shubhadeep Chakraborty Submitted By:-**

**Pappu Kumar Pashi(174060283)**

**Head Of Department**

**Department of IT**

CERTIFICATE

****

This is to certify that the Industrial Training report entitled “**VOICE ENABLED STUDENT DATABASE MANAGMENT**” Submitted by **Pappu Kumar Pashi** of Third Year B.Tech in the year 2019-2020 ofIT Department of this institute is a satisfactory account of his Industrial Training Work based on syllabus which is approved for the award of degree of Bachelor of Technology and Bachelor of Computer Application.

**Internal Examiner**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Acknowledgement

We are most obliged and grateful to our guide Mr. Shubhadeep Chakraborty Sir (Professional Trainer, Content Writter and Developer in Python, Machine Learning, Data Science) for giving us proper guidance in completing this project successfully. We are also grateful for right guidance and advice at the crucial junctures and for showing us the right way and helping us to completing this project successfully. We are most obliged and grateful to our guider for supporting and encouraging us throughout the course of study.

Finally, we acknowledge with gratitude the unflagging support and patience of our team members of their guidance and encouragement during this dissertation word.

**What is Machine Learning? A definition**

 Machine learning



Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. **The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly.

Evolution of machine learning

Because of new computing technologies, machine learning today is not like machine learning of the past. It was born from pattern recognition and the theory that computers can learn without being programmed to perform specific tasks; researchers interested in artificial intelligence wanted to see if computers could learn from data. The iterative aspect of machine learning is important because as models are exposed to new data, they are able to independently adapt. They learn from previous computations to produce reliable, repeatable decisions and results. It’s a science that’s not new – but one that has gained fresh momentum.

While many machine learning algorithms have been around for a long time, the ability to automatically apply complex mathematical calculations to [big data](https://www.sas.com/en_in/insights/big-data/what-is-big-data.html) – over and over, faster and faster – is a recent development. Here are a few widely publicized examples of machine learning applications you may be familiar with:

* The heavily hyped, self-driving Google car? The essence of machine learning.
* Online recommendation offers such as those from Amazon and Netflix? Machine learning applications for everyday life.
* Knowing what customers are saying about you on Twitter? Machine learning combined with linguistic rule creation.
* Fraud detection? One of the more obvious, important uses in our world today.

## Why is machine learning important?

Resurging interest in machine learning is due to the same factors that have made [data mining](https://www.sas.com/en_in/insights/analytics/data-mining.html) and Bayesian analysis more popular than ever. Things like growing volumes and varieties of available data, computational processing that is cheaper and more powerful, and affordable data storage.

All of these things mean it's possible to quickly and automatically produce models that can analyze bigger, more complex data and deliver faster, more accurate results – even on a very large scale. And by building precise models, an organization has a better chance of identifying profitable opportunities – or avoiding unknown risks.

### What's required to create good machine learning systems?

* Data preparation capabilities.
* Algorithms – basic and advanced.
* Automation and iterative processes.
* Scalability.
* Ensemble modeling.

### Did you know?

* In machine learning, a target is called a label.
* In statistics, a target is called a dependent variable.
* A variable in statistics is called a feature in machine learning.
* A transformation in statistics is called feature creation in machine learning.

Content

* **PACKAGES & MODULES:NUMPY**
* **PACKAGES & MODULES: PANDAS**
* **SPEECHRECOGNITION**
* **The ndarray data structure, Limitations**
* **Pandas Data Frames, Features**
* **SPEECHRECOGNITION**
* **Speech Recognition with Python**
* **TIME**
* **CASE STUDY**
* **Generic Speech Recognition System:**
* **APPLICATIONS:**
* **Replacing complicated and often frustrating ‘push button’ IVR:**
* **Medical documentation**
* **High-performance fighter aircraft**
* **Usage in education and daily life**
* **Installation**
* **USES**
* **Further applications**
* **ALGORITHM**
* **FLOW CHART**
* **Coding part of the program**
* **SCREENSHORT OF OUTPUT**

**PACKAGES & MODULES:**

**NUMPY**

**NumPy** isnotanotherprogramminglanguagebuta **Python** extensionmodule**.** Itprovidesfastandefficientoperationsonarraysofhomogeneousdata**. NumPy** extendspythonintoahigh**-**levellanguageformanipulatingnumericaldata**,** similartoMATLAB**.**It'sfree**, i.e.** itdoesn'tcostanythingandit isopensource**.** It'sanextensionon **Python** ratherthanaprogramminglanguageonitsown**.** NumPyusesPythonsyntax**.** Because **NumPy** is **Python,** embeddingcodefromotherlanguageslikeC,C++andFortranisverysimple**.**

**NumPy** (pronounced / (*NUM-py*) is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

* a powerful N-dimensional array object
* sophisticated (broadcasting) functions
* tools for integrating C/C++ and Fortran code
* useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Both NumPy and SciPy are not part of a basic Python installation. They have to be installed after the Python installation. NumPy has to be installed before installing SciPy.

### The ndarray data structure:

The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.

### Limitations:

Inserting or appending entries to an array is not as trivially possible as it is with Python's lists. The np.pad(...) routine to extend arrays actually creates new arrays of the desired shape and padding values, copies the given array into the new one and returns it. NumPy's np.concatenate([a1,a2]) operation does not actually link the two arrays but returns a new one, filled with the entries from both given arrays in sequence. Reshaping the dimensionality of an array with np.reshape(...) is only possible as long as the number of elements in the array does not change..

When we say "Core Python", we mean Python without any special modules, i.e. especially without NumPy.

The advantages of Corecomputation:

* high-level number objects: integers, floating point
* containers: lists with cheap insertion and append methods, dictionaries with fast lookup

Advantages of using Numpy with Python:

* array oriented computing
* efficiently implemented multi-dimensional arrays

**PANDAS**

***Pandas*** is an open source, BSD-licensed library providing high-performance, easy-to-use data *structures* and data analysis tools for the Python programming language.

Pandas stands for “Python Data Analysis Library”. According to the Wikipedia page on Pandas, “the name is derived from the term “panel data”, an econometrics term for multidimensional structured data sets.”

Pandas is a NumFOCUS sponsored project. This will help ensure the success of development of pandas as a world-class open-source project, and makes it possible to donate to the project.

In computer programming, **pandas**is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

Developer Wes McKinney started working on pandas in 2008 while at AQR capital management out of the need for a high performance, flexible tool to perform quantitative analysis on financial data. Before leaving AQR he was able to convince management to allow him to open source the library.

Pandas is well suited for many different kinds of data:

* Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
* Ordered and unordered (not necessarily fixed-frequency) time series data.
* Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
* Any other form of observational / statistical data sets. The data actually need not be labelled at all to be placed into a panda’s data structure.
* Flexible reshaping and pivoting of data sets
* Hierarchical labelling of axes (possible to have multiple labels per tick)
* Robust IO tools for loading data from flat files (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast HDF5 format
* Time series-specific functionality: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging, etc.

**Pandas Data Frames:**

Before you start, let’s have a brief recap of what DataFrames are.

Those who are familiar with R know the data frame as a way to store data in rectangular grids that can easily be overviewed. Each row of these grids corresponds to measurements or values of an instance, while each column is a vector containing data for a specific variable. This means that a data frame’s rows do not need to contain, but can contain, the same type of values: they can be numeric, character, logical, etc.

Now, DataFrames in Python are very similar: they come with the Pandas library, and they are defined as a two-dimensional labelled data structures with columns of potentially different types.

In general, you could say that the Pandas DataFrame consists of three main components: the data, the index, and the columns.

1. Firstly, the DataFrame can contain data that is:

* a Pandas DataFrame
* A Pandas series: a one-dimensional labelled array capable of holding any data type with axis labels or index. An example of a Series object is one column from a DataFrame.
* a NumPy ndarray, which can be a record or structured
* a two-dimensional ndarray
* **Features:**
* DataFrame object for data manipulation with integrated indexing.
* Tools for reading and writing data between in-memory data structures and different file formats.
* Data alignment and integrated handling of missing data.
* Reshaping and pivoting of data sets.
* Label-based slicing, fancy indexing, and subsetting of large data sets.
* Data structure column insertion and deletion.
* Group by engine allowing split-apply-combine operations on data sets.
* Data set merging and joining.

The library is highly optimized for performance, with critical code paths written in Cython or C.

* A fast and efficient Data Frame object for data manipulation with integrated indexing;
* Tools for reading and writing data between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format;
* Aggregating or transforming data with a powerful group by engine allowing split-apply-**SPEECHRECOGNITION**

In computer programming, **SpeechRecognition** is a software library written for the Python programming language.Speech recognition has its roots in research done at Bell Labs in the early 1950s. Early systems were limited to a single speaker and had limited vocabularies of about a dozen words. Modern speech recognition systems have come a long way since their ancient counterparts. They can recognize speech from multiple speakers and have enormous vocabularies in numerous languages.

Speech Recognition is an important feature in several applications used such as home automation, artificial intelligence, etc. This article aims to provide an introduction on how to make use of the SpeechRecognition library of Python. This is useful as it can be used on microcontrollers such as RaspberriPis with the help of an external microphone.

.

Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API,

Microsoft Bing Voice Recognition and IBM Speech to Text.

The SpeechRecognition module depends on pyaudio, you can install them from your package manager.

On Manjaro Linux these packages are called “python-pyaudio” and “python2-pyaudio”, they may have another name in your system.

In this tutorial we will use Google Speech Recognition Engine with Python.

The first component of speech recognition is, of course, speech. Speech must be converted from physical sound to an electrical signal with a microphone, and then to digital data with an analog-to-digital converter. Once digitized, several models can be used to transcribe the audio to text.

Most modern speech recognition systems rely on what is known as a Hidden Marcov model (HMM). This approach works on the assumption that a speech signal, when viewed on a short enough timescale (say, ten milliseconds), can be reasonably approximated as a stationary process—that is, a process in which statistical properties do not change over time.

The SpeechRecognition library acts as a wrapper for several popular speech APIs and is thus extremely flexible. One of these—the Google Web Speech API—supports a default API key that is hard-coded into the SpeechRecognition library. That means you can get off your feet without having to sign up for a service.

Speech Recognition with Python

This repository contains resources from [The Ultimate Guide to Speech Recognition with Python](https://realpython.com/python-speech-recognition/) tutorial on Real Python.

Audio files for the examples in the Working with Audio Files section of the post can be found in the audio files directory. To download them, use the green "Clone or download" button at the top right corner of this page.

The guessing\_game.py file contains the full source code for the "Guess a Word" game example.

**NOTE**: You will need to install the [SpeechRecognition](https://github.com/Uberi/speech_recognition) and [PyAudio](https://people.csail.mit.edu/hubert/pyaudio/) packages in orderto run the example. Please see the [tutorial](https://realpython.com/python-speech-recognition/) for step-by-step instructions.

You can test your SpeechRecognition and PyAudio installation by downloading guessing\_game.py and typing the following into a Python REPL session:

>>>importspeech\_recognitionassr

>>>from guessing\_game.py importrecognize\_speech\_from\_mic

>>> r =sr.Recognizer()

>>> m =sr.Microphone()

>>>recognize\_speech\_from\_mic(r, m) # speak after running this line

{'success': True, 'error': None, 'transcription': 'hello'}

Of course, your output will vary depending on what you said after running recognize\_speech\_from\_mic(r, m).

**TIME**

This module provides various time-related functions. For related functionality, see also the datetime and calendar modules.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

An explanation of some terminology and conventions is in order.

The epoch is the point where the time starts. On January 1st of that year, at 0 hours, the “time since the epoch” is zero. For UNIX, the epoch is 1970. To find out what the epoch is, look at gmtime (0).

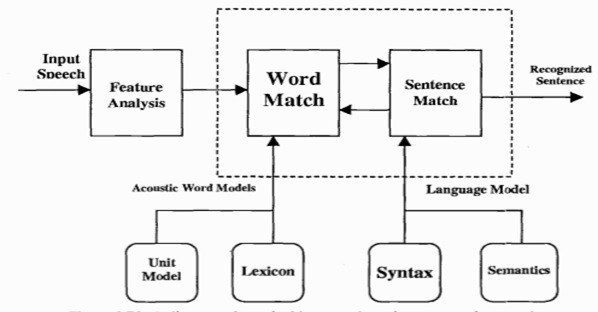
The functions in this module do not handle dates and times before the epoch or far in the future. The cut-off point in the future is determined by the C library; for UNIX, it is typically in 2038.

CASE STUDY

**Speech recognition** (also known as **voice recognition**) is the process of converting spoken words into [computer](https://simple.wikipedia.org/wiki/Computer) text. The user speaks into a [microphone](https://simple.wikipedia.org/wiki/Microphone) and the computer creates a text file of the words they have spoken.

Although the [accuracy](https://simple.wikipedia.org/wiki/Accuracy) of these systems has improved in the 21st century, they are still far from perfect. If you only need them to recognise a few words, for example the words *one, two, three* etc., then [speech](https://simple.wikipedia.org/wiki/Speech) recognition [software](https://simple.wikipedia.org/wiki/Software) can be very accurate. This is why this feature has started to appear on modern [mobile phones](https://simple.wikipedia.org/wiki/Mobile_phone). But systems that can recognise any word spoken by any person in any language are still a few years away.

This process fundamentally functions as a pipeline that converts PCM (Pulse Code Modulation) digital audio from a sound card into recognized speech. Speech recognition technology has evolved for more than 40 years, spurred on by advances in signal processing, algorithms, architectures, and hardware. During that time it has gone from a laboratory curiosity to an art, and eventually to a full-fledged technology that is practiced and understood by a wide range of engineers, scientists, linguists, psychologists, and systems designers. Over those 4 decades, the technology of speech recognition has evolved, leading to a steady stream of increasingly more difficult asks which have been tackled and solved.

[](https://krazytech.com/technical-papers/speech-recognition/attachment/integrated-continuous-speech-recognition-system)***Generic Speech Recognition System:***

The figure shows a block diagram of a typical integrated continuous speech recognition system. Interestingly enough, this generic block diagram can be made to work on virtually any speech recognition task that has been devised in the past 40 years, i.e. isolated word recognition, connected word recognition, continuous speech recognition, etc. The feature analysis module provides the acoustic feature vectors used to characterize the spectral properties of the time-varying speech signal. The word level acoustic match module evaluates the similarity between the input feature vector sequence (corresponding to a portion of the input speech) and a set of acoustic word models for all words in the recognition task vocabulary to determine which words were most likely spoken.

      Almost every aspect of the continuous speech recognizer of Figure 1 has been studied and optimized over the years. As a result, we have obtained a great deal of knowledge about how to design the feature analysis module, how to choose appropriate recognition units, how to populate the word lexicon, how to build acoustic word models, how to model language syntax and semantics, how to decode word matches against word models, how to efficiently determine a sentence match, and finally how to eventually choose the best recognized sentence.

*APPLICATIONS:*

**Automation of Operator Services**Systems like the Voice Recognition Call Processing (VRCP) system introduced by AT&T or the Automated Alternate Billing System (AABS) introduced by Nortel enabled operator functions to be handled by speech recognition systems. The VRCP system handled so-called ‘operator assisted’ calls such as Collect, Third Party Billing, and Person-to-Person, Operator Assisted Calling and Calling Card calls. The AABS system automated the acceptance (or rejection) of billing charges for reverse calls by recognizing simple variants of the two-word vocabulary Yes and No.

**Automation of Directory Assistance** Systems was created for assisting operators with the task of determining telephone numbers in response to customer queries by voice. Both NYNEX and Nortel introduced a system that did front end city name recognitionJ so as to reduce the operator search space for the desired listing, and several experimental systems were created to complete the directory assistance task by attempting to recognize individual names in a directory of as many as 1 million names. Such systems are not yet practical (because of the confusability among names) but for small directories, such systems have been widely used (e.g., in corporate environments).

**Voice Dialing** Systems have been created for voice dialing by name (so-called alias dialing such as Call Home, Call Office) from AT&T, NYNEX, and Bell Atlantic, and by number (AT&T SDN/NRA) to enable customers to complete calls without having to push buttons associated with the telephone number being called.

### ****Replacing complicated and often frustrating ‘push button’ IVR:****

Due to poorly implemented and managed systems, IVR and automated call handling systems may be often unpopular and frustrating with customers. However, there is a way to improve this scenario. Termed ‘intelligent call steering’ (ICS), it does not involve any ‘button pushing’. The system simply asks the customer what they want (in their words, not yours) and then transfers them to the most suitable resource to handle their call. Callers dial one number and are greeted by the message “Welcome to XYZ Company, how I can help you?” The caller is routed to the right agent within 20 to 30 seconds of the call being answered with misdirected calls reduced to as low as 3-5 percent.

By introducing Natural Language Speech Recognition (NLSR), general insurance company Suncorp replaced its original push button IVR, enabling the customer to simply say what they want. Using a financial services’ statistical language model of over 100,000 phrases, the system can more accurately assess the nature of the call and transfer it the first time to the appropriate department or advisor. The company reduced its call waiting times to around 30 seconds and misdirected calls to virtual nil.

### ****In-car systems****

      Typically a manual control input, for example by means of a finger control on the steering wheel, enables the speech recognition system and this is signaled to the driver by an audio prompt. Following the audio prompt, the system has a “listening window” during which it may accept a speech input for recognition.

      Simple voice commands may be used to initiate phone calls, select radio stations or play music from a compatible smartphone, MP3 player or music-loaded flash drive. Voice recognition capabilities vary between car make and model. Some of the most recent car models offer natural-language speech recognition in place of a fixed set of commands, allowing the driver to use full sentences and common phrases. With such systems, there is, therefore, no need for the user to memorize a set of fixed command words.

### ****Medical documentation****

   In the health care sector, speech recognition can be implemented in front-end or back-end of the medical documentation process. Front-end speech recognition is where the provider dictates into a speech-recognition engine, the recognized words are displayed as they are spoken, and the dictator is responsible for editing and signing off on the document. Back-end or deferred speech recognition is where the provider dictates into a digital dictation system, the voice is routed through a speech-recognition machine and the recognized draft document is routed along with the original voice file to the editor, where the draft is edited and report finalized. Deferred speech recognition is widely used in the industry currently.

### ****High-performance fighter aircraft****

Substantial efforts have been devoted in the last decade to the test and evaluation of speech recognition in fighter aircraft. Of particular note have been the US program in speech recognition for the Advanced Fighter Technology Integration (AFTI)/F-16 aircraft (F-16 VISTA), the program in France for Mirage aircraft, and other programs in the UK dealing with a variety of aircraft platforms. In these programs, speech recognizers have been operated successfully in fighter aircraft, with applications including setting radio frequencies, commanding an autopilot system, setting steer-point coordinates and weapons release parameters, and controlling flight display.

### ****Usage in education and daily life****

For language learning, speech recognition can be useful for learning a second language. It can teach proper pronunciation, in addition to helping a person develop fluency with their speaking skills.

Students who are blind (see Blindness and education) or have very low vision can benefit from using the technology to convey words and then hear the computer recite them, as well as use a computer by commanding with their voice, instead of having to look at the screen and keyboard.

Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API,  
Microsoft Bing Voice Recognition and IBM Speech to Text.

## Installation

A library that helps is named “SpeechRecognition”. You should install it with pyenv, pipenv or virtualenv. You can also install it system wide:

|  |
| --- |
| pip install SpeechRecognition |

The SpeechRecognition module depends on pyaudio, you can install them from your package manager.  
On Manjaro Linux these packages are called “python-pyaudio” and “python2-pyaudio”, they may have another name in your system.

**Speech Recognition demo**  
You can test the speech recognition module, with the command:

python -m speech\_recognition

Results show in terminal.

**Speech Recognition with Google**  
The example below uses Google Speech Recognition engine, which I’ve tested for the English language.

For testing purposes, it uses the default API key.  
To use another API key, use

|  |
| --- |
| `r.recognize\_google(audio, key="GOOGLE\_SPEECH\_RECOGNITION\_API\_KEY")` |

Copy the code below and save the file as speechtest.py.  
Run it with Python 3.

|  |
| --- |
| #!/usr/bin/env python3    importspeech\_recognitionassr    # get audio from the microphone  r = sr.Recognizer()  withsr.Microphone() as source:  print("Speak:")  audio = r.listen(source)    try:  print("You said " + r.recognize\_google(audio))  exceptsr.UnknownValueError:  print("Could not understand audio")  exceptsr.RequestErroras e:  print("Could not request results; {0}".format(e)) |

In our project we have tried to create a database using speech recognition. By using our program one can create a database, modify a database fully or partially modify a database .this program can be useful everywhere in schools, colleges and even in offices . In schools and colleges teachers can create the database the database of thousands of students fully by using speech that is just by speaking and not by typing for hours which reduces their labour. Even in offices the database of an employees can also be created using speech recognition. In short it will bring an evolution in the reception counters of schools, colleges and offices.

USES

Used in evolving search engines; when using search engines there can be differences between how we type our inquiries and how we verbalize the same queries. The user may have trouble expressing a phrase or their intent thus may not acquire appropriate results. With the inclusion of speech recognition in search engines, the results accuracies will be significantly increased. As speech recognition improves, there will be a significant implication on how the public views search engines generally.

Impact in the healthcare industry; the feature has its use in medical reporting by medical personnel. When it was introduced in this industry doctors had trouble using it to accomplish tasks. The system had a limited understanding of medical terminologies. Therefore, doctors had to learn on how to talk to the software. The technology was improved to be user-friendly and accurate; this was established by imperative improvements and inclusion of relevant vocabularies.

Use in service delivery; customers and clients may not want to speak to a live operator. Therefore, they opt to use the speech recognition systems. This helps to make the process efficient and improves on time as it cuts on waiting time. This has its application in various airports in confirming travel schedules of the aircraft.

Communication in service providers; telecommunication providers use speech recognition to serve their clients who may want to receive customer care services. This consists of various questions by the software to establish the caller’s demands and then directs them to the appropriate operator for assistance.

**Further applications**

* Aerospace (e.g. space exploration, spacecraft, etc.) NASA's Mars Polar Lander used speech recognition technology from Sensory, Inc. in the Mars Microphone on the Lander
* Automatic subtitling with speech recognition
* Automatic translation
* Court reporting (Realtime Speech Writing)
* recognition computer user interface
* Home automation
* Interactive voice response
* Mobile telephony, including mobile email
* Multimodal interaction
* Pronunciation evaluation in computer-aided language learning applications
* Robotics

ALGORITHM

1. In this project we tried to make a system which is not manually accessible and here we used the speech Recognition system to implement this. At first we defined some functions through which we can store the data.

2. Here we used four packages like Speech Recognition, time, pandas, numpy. At first we declared some empty list. And we imported the packages in the beginning of the program.

3. In this program we used Google API to use the speech Recognition function.

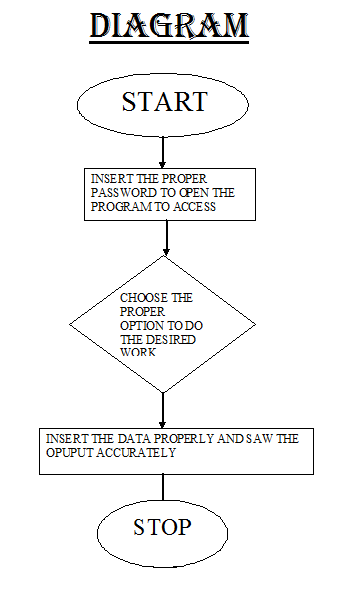
4. In the first function we defined the speech Recognition. Then we defined the function from where we need to select the function we want to perform like we want to add data or delete data or upgrade the database or we want to create data base or not.

5. After that we defined all the functions separately. And we used the functions in the main body of the program.

6. In the main program we initialized the coding lines through which user can get a clear idea about what to do.

7. We used a password protection for the access of the datasheet. And there is an option to lock datasheet after the three times try.

FLOW CHART



Coding part of the program

import time

import numpy as np

import pandas as pd

import speech\_recognition as sr

rlno=[]

name=[]

dob=[]

stream=[]

year=[]

grade=[]

def speechip():

r = sr.Recognizer()

with sr.Microphone() as source:

audio = r.listen(source)

try:

print("You said: " + r.recognize\_google(audio))

except sr.UnknownValueError:

print("Google Speech Recognition could not understand audio")

return

except sr.RequestError as e:

print("Could not request results from Google Speech Recognition service; {0}".format(e))

return

speech=r.recognize\_google(audio)

return speech

def options():

#ch=input("Enter into operation(y/Y or n/N): ")

print('speak "YES" to continue "NO" to discontinue')

ch=speechip()

while(ch=='yes' or ch=='YES'):

#v=int(input("Enter your Action:"))

print('speak:Action-1: Create Database \nAction-2. Modify Database \nAction-3. Delete Database')

v=speechip()

if v=="Create Database" or v=="create database":

createDatabase()

elif v=="Modify Database" or v=="modify database":

modifyDatabase()

elif v=="Delete Database" or v=="delete database":

deleteDatabase()

else:

exit(0)

#ch=input("Continue?(y/Y or n/N): ")

print('Continue? speak "YES" or"NO"')

ch=speechip()

def modifyDatabase():

print("1. Entire Modification \n2. Partial Modification \n3. Exit \n:")

#ch1=int(input("Enter Choice:"))

print('speak your choice:')

ch1=speechip()

if ch1==1:

entireModify()

#options()

elif ch1==2:

partialModify()

options()

else:

options()

def insertdata(rlno,name,dob,stream,year,grade):

#Creating database in csv file in same location

filedf=pd.DataFrame({

'UNIVERSITY ROLL NUMBER': rlno,

'NAME': name,

'D.O.B': dob,

'STREAM': stream,

'YEAR': year,

'GRADE': grade

})

#print(filedf)

#location of the csv file

filedf.to\_csv("C:/Users/PAPPU KUMAR/Student's Database management using Speech recognition/Student's Database management using Speech recognition/StudentData.csv")

return

def createDatabase():

#ch=input("Insert data?(y/Y or n/N): ")

print('Insert data: speak YES or NO')

ch=speechip()

while(ch=='yes' or ch=='YES'):

print("Speak UNIVERSITY ROLL NUMBER: ")

rl=speechip()

rlno.append(rl)

print("Speak Name: ")

nm=speechip()

name.append(nm)

print("Speak Date of Birth: ")

db=speechip()

dob.append(db)

print("Speak Stream: ")

st=speechip()

stream.append(st)

print("Speak Year: ")

yr=speechip()

year.append(yr)

print("Speak Grade: ")

gd=speechip()

grade.append(gd)

insertdata(rlno,name,dob,stream,year,grade)

#ch=input("Continue?(y/Y or n/N): ")

print('Continue: speak YES or NO')

ch=speechip()

options()

def entireModify():

#n=input("Enter UNIVERSITY ROLL NUMBER to Modify respective database:")

print('Enter UNIVERSITY ROLL NUMBER to Modify respective database:')

n=speechip()

file=pd.read\_csv("C:/Users/PAPPU KUMAR/Student's Database management using Speech recognition/Student's Database management using Speech recognition/StudentData.csv")

namear=np.array(file['UNIVERSITY ROLL NUMBER'].tolist())

namear.flatten()

namelist=namear.tolist()

ind=namelist.index(n)

#modpt=file.ix[nameind]

#modlist=modpt.tolist()

#Process

#ch=input("Insert data?(y/Y or n/N): ")

print('Insert data:speak YES or NO')

ch=speechip()

while(ch=='yes' or ch=='YES'):

print("Speak UNIVERSITY ROLL NUMBER: ")

rl=speechip()

rlno[ind]=rl

print("Speak Name: ")

nm=speechip()

name[ind]=nm

print("Speak Date of Birth: ")

db=speechip()

dob[ind]=db

print("Speak Stream: ")

st=speechip()

stream[ind]=st

print("Speak Year: ")

yr=speechip()

year[ind]=yr

print("Speak Grade: ")

gd=speechip()

grade[ind]=gd

insertdata(rlno,name,dob,stream,year,grade)

#ch=input("Continue?(y/Y or n/N): ")

print('Continue:speak YES or NO')

ch=speechip()

options()

def partialModify():

#n=input("Enter UNIVERSITY ROLL NUMBER to Modify respective database:")

print('Enter UNIVERSITY ROLL NUMBER to Modify respective database:')

n=speechip()

file=pd.read\_csv("C:/Users/PAPPU KUMAR/Student's Database management using Speech recognition/Student's Database management using Speech recognition/StudentData.csv")

namelist=file['UNIVERSITY ROLL NUMBER'].tolist()

#print(namear)

#namear.flatten()

ind=namelist.index(n)

print(namelist)

print(ind)

#modpt=file.ix[nameind]

#modlist=modpt.tolist()

#ch2=input("Custom Modification, (y/Y or n/N)?")

print('Custom Modification:speak YES or NO?')

ch2=speechip()

while ch2=='yes' or ch2=='YES':

#sect=int(input("1. UNIVERSITY ROLL NUMBER\n2. Name \n3. DOB \n4. Stream \n5. Year \n6. Grade \n:"))

print('speak 1. UNIVERSITY ROLL NUMBER\n2. Name \n3. DOB \n4. Stream \n5. Year \n6. Grade \n: ')

sect=speechip()

if sect==1:

print("Speak UNIVERSITY ROLL NUMBER: ")

rl=speechip()

print(rlno)

rlno[ind]=rl

print(rlno)

elif sect==2:

print("Speak Name: ")

nm=speechip()

name[ind]=nm

elif sect==3:

print("Speak Date of Birth: ")

db=speechip()

dob[ind]=db

elif sect==4:

print("Speak Stream: ")

st=speechip()

stream.append(st)

elif sect==5:

print("Speak Year: ")

yr=speechip()

year.append(yr)

elif sect==6:

print("Speak Grade: ")

gd=int(speechip())

grade.append(gd)

insertdata(rlno,name,dob,stream,year,grade)

#ch=input("Continue?(y/Y or n/N): ")

print('Continue:speak YES or NO?')

ch=speechip()

return

def deleteDatabase():

#n=input("Enter UNIVERSITY ROLL NUMBER to Delete respective database:")

print('Enter UNIVERSITY ROLL NUMBER to Delete respective database:')

n=speechip()

file=pd.read\_csv("C:/Users/PAPPU KUMAR/Student's Database management using Speech recognition/Student's Database management using Speech recognition/StudentData.csv")

namear=np.array(file['UNIVERSITY ROLL NUMBER'].tolist())

namear.flatten()

namelist=namear.tolist()

ind=namelist.index(n)

print(ind)

#ch2=input("Custom Modification, (y/Y or n/N)?")

print('Custom Modification:YES or NO?')

ch2=speechip()

col=[]

r1=file['UNIVERSITY ROLL NUMBER'].tolist()

n1=file['NAME'].tolist()

d1=file['D.O.B'].tolist()

s1=file['STREAM'].tolist()

y1=file['YEAR'].tolist()

g1=file['GRADE'].tolist()

while ch2=='yes' or ch2=='YES':

del r1[ind]

del n1[ind]

del d1[ind]

del s1[ind]

del y1[ind]

del g1[ind]

rlno=r1

name=n1

dob=d1

stream=s1

year=y1

grade=g1

insertdata(rlno,name,dob,stream,year,grade)

#ch=input("Continue?(y/Y or n/N): ")

print('Continue:YES or NO?')

ch=speechip()

return

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~PROJECT~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")

print("~~~~~~~~~~~~~Welcome to Smart Voice Control Database Management System~~~~~~~~~~~~")

for i in range(4):

if i==3:

print("You have exceeded maximum try for",i,"times !!! Sorry!!")

break

print("!!! Please Speak password to open Database of Students!!!")

password=speechip()

if password=='project':

#print("This will: \nAction-1: Create Database \nAction-2. Modify Database \nAction-3. Delete Database ")

print('Go to the option window')

options()

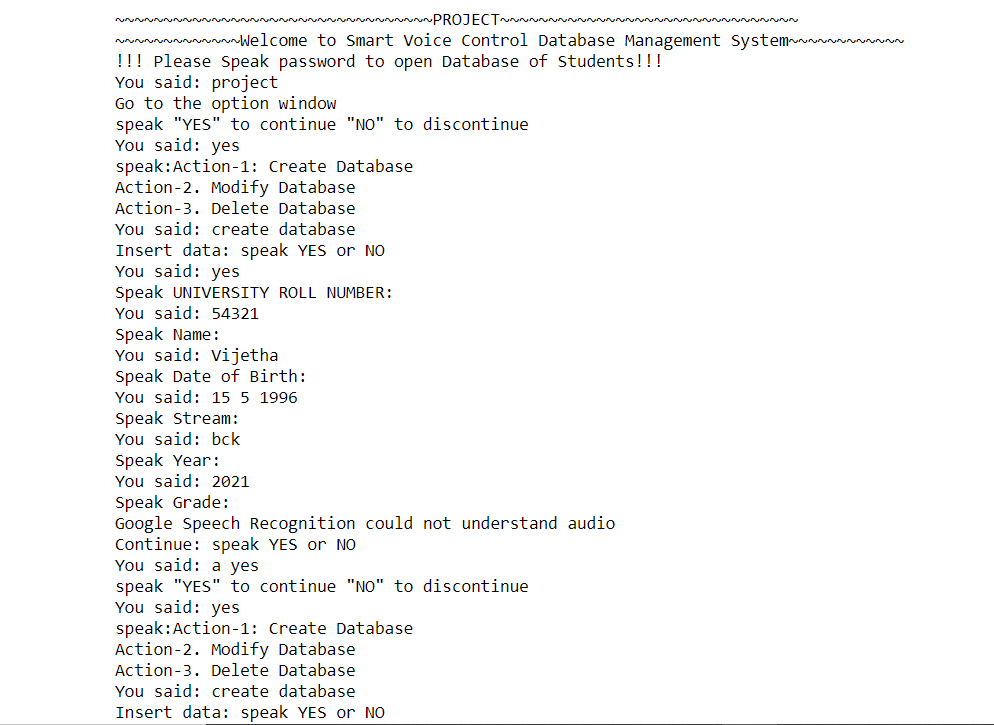
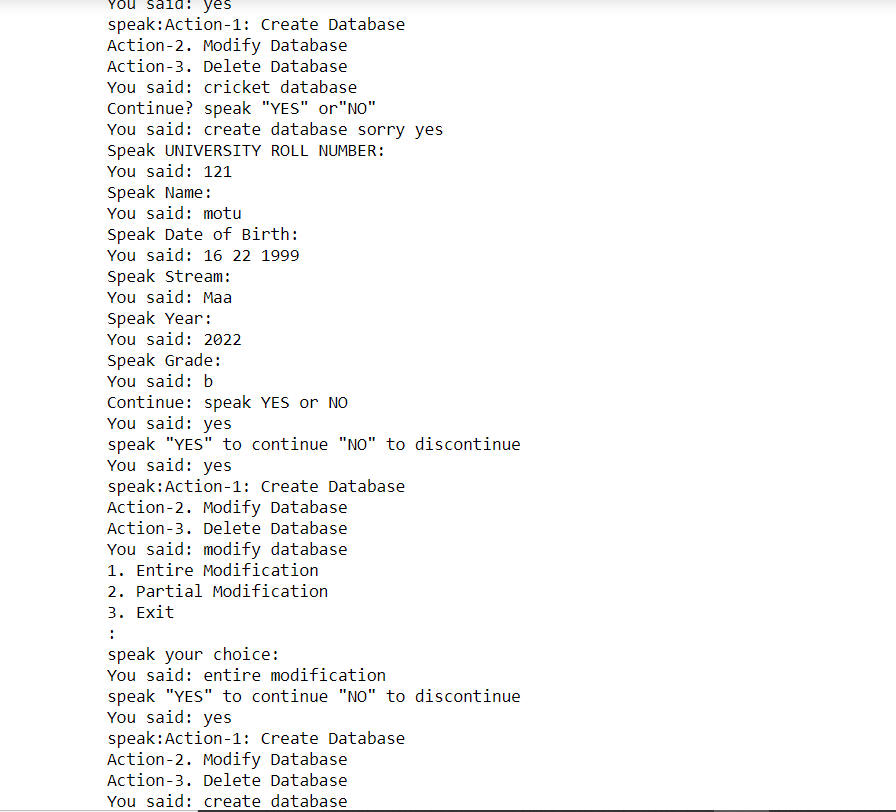
else:

print("Password Error!!!!! Try again!!!!!")

filedf=pd.read\_csv("C:/Users/PAPPU KUMAR/Student's Database management using Speech recognition/Student's Database management using Speech recognition/StudentData.csv")

r,c=filedf.shape

filedf.head(r)

SCREENSHORT OF OUTPU