**CREATE A VOICE CHATBOT**

# **Using DATA TYPES, LIBRARIES AND FUNCTIONS of Python**

## **Data Types**

|  |  |
| --- | --- |
| **Text Type:** | **str** |
| **Numeric Types:** | **int, float, complex** |
| **Sequence Types:** | **list, tuple, range** |
| **Mapping Type:** | **dict** |
| **Set Types:** | **set, frozenset** |
| **Boolean Type:** | **bool** |
| **Binary Types:** | **bytes, bytearray, memoryview** |
| **None Type:** | **NoneType** |

## **LIBRARIES**

* [**TensorFlow**](https://www.edureka.co/blog/python-libraries/#z1)
* [**Scikit-Learn**](https://www.edureka.co/blog/python-libraries/#z2)
* [**Numpy**](https://www.edureka.co/blog/python-libraries/#z3)
* [**Keras**](https://www.edureka.co/blog/python-libraries/#z4)
* [**PyTorch**](https://www.edureka.co/blog/python-libraries/#z5)
* [**LightGBM**](https://www.edureka.co/blog/python-libraries/#z6)
* [**Eli5**](https://www.edureka.co/blog/python-libraries/#z7)
* [**SciPy**](https://www.edureka.co/blog/python-libraries/#z8)
* [**Theano**](https://www.edureka.co/blog/python-libraries/#z9)
* [**Pandas**](https://www.edureka.co/blog/python-libraries/#z10)
* **NuPIC**
* **Ramp**
* **Pipenv**
* **PyBrain**
* **Matplotlib**

## **Functions**

* **Built-in Functions**
* **User-defined Functions**
* **Recursive Functions**
* **Lambda Function.**

pip install SpeechRecognition

pip install gtts

import speech\_recognition as sr

from gtts import gTTS

import os

def listen\_to\_user():

recognizer = sr.Recognizer()

with sr.Microphone() as source:

print("Say something:")

recognizer.adjust\_for\_ambient\_noise(source)

audio = recognizer.listen(source)

try:

user\_input = recognizer.recognize\_google(audio).lower()

print("You said:", user\_input)

return user\_input

except sr.UnknownValueError:

print("Sorry, I could not understand what you said.")

return ""

except sr.RequestError as e:

print("Error connecting to Google API: {0}".format(e))

return ""

def respond\_to\_user(response\_text):

tts = gTTS(text=response\_text, lang='en')

tts.save("response.mp3")

os.system("start response.mp3")

def chatbot():

while True:

user\_input = listen\_to\_user()

if user\_input == 'exit':

print("Goodbye!")

break

elif user\_input:

# Here, you can implement your chatbot logic based on user\_input

response\_text = "I heard you say: " + user\_input

respond\_to\_user(response\_text)

if \_\_name\_\_ == "\_\_main\_\_":

print("Voice Chatbot: Hello! How can I assist you today?")

chatbot()

----------------------------------------------------------------------------------------------------------------

pip install SpeechRecognition pyttsx3

import speech\_recognition as sr

import pyttsx3

def listen():

recognizer = sr.Recognizer()

with sr.Microphone() as source:

print("Listening...")

recognizer.adjust\_for\_ambient\_noise(source)

audio = recognizer.listen(source)

try:

print("Recognizing...")

user\_input = recognizer.recognize\_google(audio)

print("You said:", user\_input)

return user\_input.lower()

except sr.UnknownValueError:

print("Sorry, I could not understand what you said.")

return ""

except sr.RequestError as e:

print("Error connecting to Google API: {0}".format(e))

return ""

def speak(response\_text):

engine = pyttsx3.init()

engine.say(response\_text)

engine.runAndWait()

def chatbot():

print("Voice Chatbot: Hello! How can I assist you today?")

while True:

user\_input = listen()

if user\_input == 'exit':

print("Voice Chatbot: Goodbye!")

speak("Goodbye!")

break

elif user\_input:

# Implement your chatbot logic based on user\_input

response\_text = "You said: " + user\_input

print("Voice Chatbot:", response\_text)

speak(response\_text)

if \_\_name\_\_ == "\_\_main\_\_":

chatbot()

----------------------------------------------------------------------------------------------------------------

import speech\_recognition as sr

import pyttsx3

def listen():

recognizer = sr.Recognizer()

with sr.Microphone() as source:

print("Listening...")

recognizer.adjust\_for\_ambient\_noise(source)

audio = recognizer.listen(source)

try:

print("Recognizing...")

user\_input = recognizer.recognize\_google(audio)

print("You said:", user\_input)

return user\_input.lower()

except sr.UnknownValueError:

print("Sorry, I could not understand what you said.")

return ""

except sr.RequestError as e:

print("Error connecting to Google API: {0}".format(e))

return ""

def speak(response\_text):

engine = pyttsx3.init()

engine.say(response\_text)

engine.runAndWait()

def marine\_management\_chatbot(user\_input):

# Implement marine management chatbot logic based on user\_input

if "fish species" in user\_input:

response\_text = "I can provide information about different fish species in your marine environment."

elif "fishing activity" in user\_input:

response\_text = "I can help you with details about recent fishing activities and catches."

elif "marine habitats" in user\_input:

response\_text = "I can provide information about various marine habitats and their conservation status."

else:

response\_text = "I'm sorry, I didn't understand your request. Please ask about fish species, fishing activities, or marine habitats."

print("Marine Management Chatbot:", response\_text)

speak(response\_text)

def chatbot():

print("Marine Management Voice Chatbot: Hello! How can I assist you today?")

while True:

user\_input = listen()

if user\_input == 'exit':

print("Marine Management Voice Chatbot: Goodbye!")

speak("Goodbye!")

break

elif user\_input:

marine\_management\_chatbot(user\_input)

if \_\_name\_\_ == "\_\_main\_\_":

chatbot()