INT 213 PYTHON PROGRAMMING

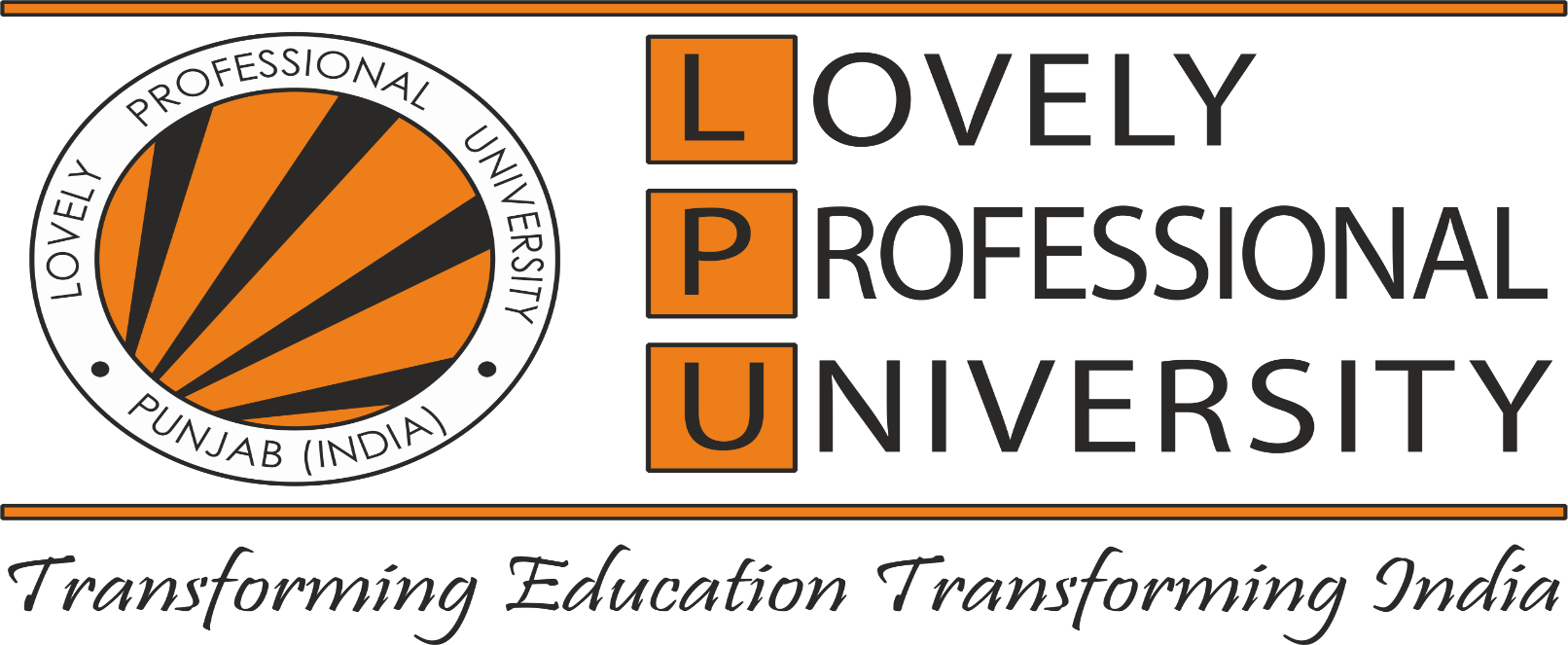
CA2 PROJECT SYNOPSIS

OF

A desktop software for helping Flood affected people to be used at flood relief camp.

Submitted by:

1. Soham Nandi, 11806959, roll no 16, K18RD
2. Roy Dev Preyo,11804924, roll no 28, K18RD



Abstract

Chatbots are being made to ease the pain that the industries are facing today. The purpose of chat bots is to support and scale business teams in their relations with customers. It could live in any major chat applications like Facebook Messenger, Slack, Telegram, Text Messages, etc.

Chatbots, or conversational interfaces as they are also known, present a new way for individuals to interact with computer systems. Traditionally, to get a question answered by a software program involved using a search engine or filling out a form. A chatbot allows a user to simply ask questions in the same manner that they would address a human. The most well-known chatbots currently are voice chatbots: Alexa and Siri. However, chatbots are currently being adopted at a high rate on computer chat platforms.

The technology at the core of the rise of the chatbot is natural language processing (“NLP”). Recent advances in machine learning have greatly improved the accuracy and effectiveness of natural language processing, making chatbots a viable option for many organizations. This improvement in NLP is firing a great deal of additional research which should lead to continued improvement in the effectiveness of chatbots in the years to come.

A simple chatbot can be created by loading an FAQ (frequently asked questions) into chatbot software. The functionality of the chatbot can be improved by integrating it into the organization’s enterprise software, allowing more personal questions to be answered, like “What is my balance?”, or “What is the status of my order?”.

Most commercial chatbots are dependent on platforms created by the technology giants for their natural language processing. These include Amazon Lex, Microsoft Cognitive Services, Google Cloud Natural Language API, Facebook DeepText, and IBM Watson. Platforms where chatbots are deployed include Facebook Messenger, Skype, and Slack, among many others.

Applications of chatbots

A chatbot can be used anywhere a human is interacting with a computer system. These are the areas where the fastest adoption is occurring:

**Customer Service**

A chatbot can be used as an “assistant” to a live agent, increasing the agent’s efficiency. When trained, they can also provide service when the call centre is closed, or eventually even act as an independent agent, if desired.

**Sales/Marketing/Branding —** Chatbots can be used for sales qualification, ecommerce, promotional campaigns, or as a branding vehicle.

**Human Resources —** An HR chatbot can help with frequently asked questions (“how many vacation days do I have left?”) and can act as an onboarding assistant

### **Accessible anytime:**

On an average people spend around 7 minutes until they are assigned to a person. Gone are the frustrating days of waiting in a queue for the next available operative. They are replacing live chat and other forms of slower contact methods such as emails and phone calls.

Since chatbots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring taking a break. This improves your customer UX and helps you rank highly in your sector. Another advantage of this instant response is that you can also skilfully craft your chatbot to maintain your image and brand.

### **Personal Assistant:**

People could use Bots as personal fashion advisor for clothing recommendations, or ask trading tips from a finance bot, suggest places to visit from a travel bot and so forth. This would help the users get a more personal touch from the chatbot. Also, the chatbot will remember all your choices and provide you with relevant choices the next time you visit it.

Notable examples are

Trim, a personal finance bot;

Taylor — travel assistant,

CNN bot for personalized news.

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.

Introduction

Our project mainly deals with machine learning algorithm. Our Chatbot is a daily user-friendly tool for LPU students. It can small talk with the student. Small talk includes daily conversion and queries about the university, and it will be also to do exceptional handling if the question asked by the user in not in our database.

It will also able to do basic mathematical calculations like add, subtract, multiplication and divide.

Also, it includes a weather API which tells about the current weather and works as travel guide.

Other features in our chatbot:

--It has voice as input as well as output

--Also, it will have graphical user interface both online and offline using tkinter and fbchat module (Facebook chat messenger).

-- Also, accuracy of our project will be higher and expected to be above 50%.

Exceptional handling:  
If a user enters any query or question that is not in our database then our chatbot can store that query or question into the database

Role of team members:

Chatbot is built and improvised by two people named Soham Nandi and Roy Dev Preyo.

Contributions:

Soham Nandi was responsible for output as voice from text. He made the offline GUI using tkinter module. Weather calling API algorithm is developed by him.

Roy Dev Preyo was responsible for input as text from voice. He made online GUI using FB chat module. Exceptional handling algorithm is developed by him.

Other than that, Basic logic and complete algorithm was developed together. Database was built and necessary modifications was made by both.

Modules/library Used

Our chatbot includes many modules and library such as

#chatterbot (for basic stemming of the words and I/O)

#pyttsx3 (for voice as input or text to speech)

# google.cloud.speech (for voice as output or speech to text)

# tkinter (for offline GUI)

#fbchat (for online GUI)

# Openweather API (for calling weather API)

**ChatterBot:**

ChatterBot is a Python library that makes it easy to generate automated responses to a user’s input. ChatterBot uses a selection of machine learning algorithms to produce different types of responses. This makes it easy for developers to create chat bots and automate conversations with users.

# How ChatterBot Works?

ChatterBot is a Python library designed to make it easy to create software that can engage in conversation.

An untrained instance of ChatterBot starts off with no knowledge of how to communicate. Each time a user enters a statement, the library saves the text that they entered and the text that the statement was in response to. As ChatterBot receives more input the number of responses that it can reply and the accuracy of each response in relation to the input statement increase.

The program selects the closest matching response by searching for the closest matching known statement that matches the input, it then chooses a response from the selection of known responses to that statement.

ChatterBot has natural language evaluation capabilities that allow it to process and evaluate **mathematical and time-based inputs**.

from chatterbot import ChatBot

bot = ChatBot(‘Math & Time Bot',

logic\_adapters= [ 'chatterbot.logic.MathematicalEvaluation', 'chatterbot.logic.TimeLogicAdapter’])

# Print an example of getting one math-based response

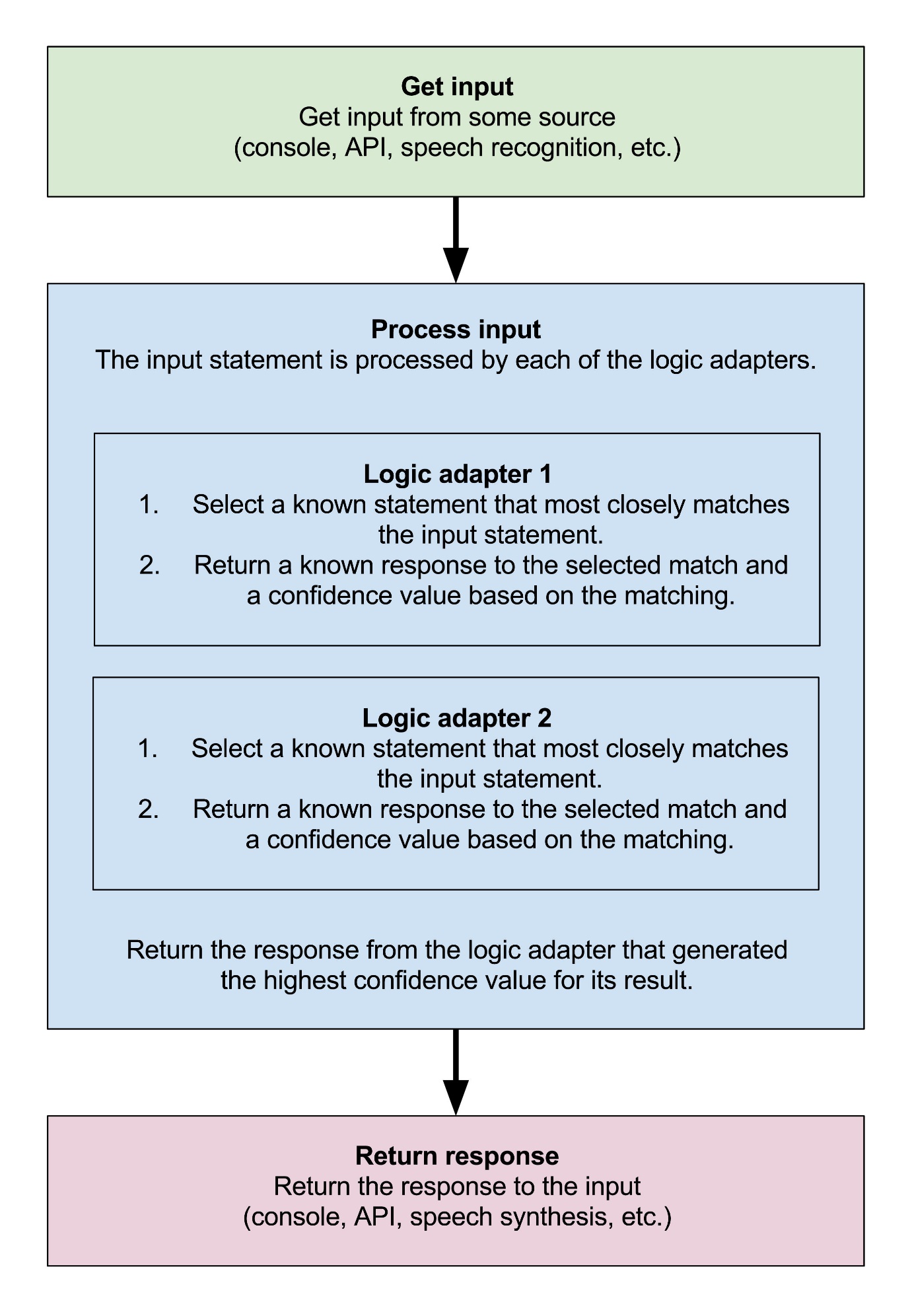
response = bot.get\_response('What is 4 + 9?') print(response)

# Print an example of getting one-time based response

response = bot.get\_response ('What time is it?') print(response)

For more details about the ideas and concepts behind ChatterBot see the process flow diagram.

# Process flow diagram



**Pyttsx for python3 [ Offline text to speech]**

Pyttsx is a good text to speech conversion library in python but it was written only in python2 until now! Even some fair amount of googling didn’t help much to get a tts library compatible with Python3.

There is however, one library **gTTS** which works perfectly in python3 but it needs internet connection to work since it relies on google to get the audio data. But Pyttsx is completely offline and works seamlessly and has multiple tts-engine support. The codes in this repo are slightly modified version of the pyttsx module of python 2.x and is a clone from westonpace’s repo. The purpose of creating this repo is to help those who want to have an offline tts lib for Python3 and don’t want to port it from python2 to python3 themselves.

This project has been dead for over some years now and pip install pyttsx doesn’t download the fixed python3 version of the library. So i decided to go ahead with my own repository and spread the fixed version.

## Usage:

import pyttsx3;

engine = pyttsx3.init();

engine.say("I will speak this text");

engine.runAndWait();

**google.cloud.speech\_v1**

The [Cloud Speech API](https://cloud.google.com/speech) enables developers to convert audio to text by applying powerful neural network models. The API recognizes over 80 languages and variants, to support your global user base.

### installation

Install this library in a [virtualenv](https://virtualenv.pypa.io/en/latest/) using pip. [virtualenv](https://virtualenv.pypa.io/en/latest/) is a tool to create isolated Python environments. The basic problem it addresses is one of dependencies and versions, and indirectly permissions.

With [virtualenv](https://virtualenv.pypa.io/en/latest/), it’s possible to install this library without needing system install permissions, and without clashing with the installed system dependencies.

**Example usage**

from google.cloud import speech\_v1

from google.cloud.speech\_v1 import enums

client = speech\_v1.SpeechClient()

encoding = enums.RecognitionConfig.AudioEncoding.FLAC

sample\_rate\_hertz = 44100

language\_code = 'en-US'

config = {'encoding': encoding, 'sample\_rate\_hertz': sample\_rate\_hertz, 'language\_code': language\_code}

uri = 'gs://bucket\_name/file\_name.flac'

audio = {'uri': uri}

response = client.recognize(config, audio)

# **Python GUI – tkinter**

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.  
**To create a tkinter:**

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

There are two main methods used you the user need to remember while creating the Python application with GUI.

 **Tk(screenName=None,  baseName=None,  className=’Tk’,  useTk=1):** To create a main window, tkinter offers a method ‘Tk(screenName=None,  baseName=None,  className=’Tk’,  useTk=1)’. To change the name of the window, you can change the className to the desired one. The basic code used to create the main window of the application is:

m=tkinter.Tk() where m is the name of the main window object

 **mainloop():** There is a method known by the name mainloop() is used when you are ready for the application to run. mainloop() is an infinite loop used to run the application, wait for an event to occur and process the event till the window is not closed.

m.mainloop ()

For more information and reference, visit Geeks for Geeks:

<https://www.geeksforgeeks.org/python-gui-tkinter/>

# **fbchat: Facebook Chat (Messenger) for Python**

Facebook Chat for Python. This project was inspired by [facebook-chat-api](https://github.com/Schmavery/facebook-chat-api).

**No XMPP or API key is needed**. Just use your email and password.

fbchat works by emulating the browser. This means doing the exact same GET/POST requests and tricking Facebook into thinking it’s accessing the website normally. Therefore, this API requires the credentials of a Facebook account.

fbchat uses your email and password to communicate with the Facebook server. That means that you should always store your password in a separate file, in case e.g. someone looks over your shoulder while you’re writing code. You should also make sure that the file’s access control is appropriately restrictive

for more information and reference, visit:

<https://fbchat.readthedocs.io/en/stable/>

References Used:

--For Basic Programming and Algorithm, we used

#python documentation

#Geeks for Geeks

#YouTube for several doubts

#Stack Overflow

--For ChatterBot Library:

#Official Documentation

<https://chatterbot.readthedocs.io/en/stable/>

--For fbchat Library:

#Official Documentation

<https://fbchat.readthedocs.io/en/stable/>

--For tkinter Library:

#Official Documentation

<https://docs.python.org/2/library/tkinter.html>

# Geeks for Geeks

<https://www.geeksforgeeks.org/python-gui-tkinter/>

--For **google.cloud library:**

**#Official Documentation:**

<https://realpython.com/python-speech-recognition/>

#Real python:

<https://realpython.com/python-speech-recognition/>

#Geeks for Geeks:

<https://www.geeksforgeeks.org/speech-recognition-in-python-using-google-speech-api/>